UNIVERSITÄT
KOBLENZ · LANDAU
Institut für Softwaretechnik

FB 4
Informatik

A First Proposal for an Overall Structure of an Enhanced Reality Framework

Jürgen Ebert
Kerstin Falkowski

Nr. 8/2007

Arbeitsberichte aus dem Fachbereich Informatik

Universität Koblenz-Landau • Campus Koblenz • Universitätsstraße 1 • 56070 Koblenz
Telefon +49 261 287-0 • http://www.uni-koblenz.de

Arbeitsberichte des Fachbereichs Informatik

ISSN (Print): 1864-0346
ISSN (Online): 1864-0850

Herausgeber / Edited by:

Der Dekan:
Prof. Dr. Paulus

Kontaktdaten der Verfasser

Jürgen Ebert, Kerstin Falkowski
Institut für Softwaretechnik
Fachbereich Informatik
Universität Koblenz-Landau
Universitätsstraße 1
D-56070 Koblenz
EMail: ebert@uni-koblenz.de

A First Proposal for an Overall Structure of an Enhanced Reality Framework

Jürgen Ebert, Kerstin Falkowski
{ebert|falke}@uni-koblenz.de
University Koblenz-Landau, Institute for Software Technology

Contents

List of Figures

1 Introduction

Enhanced Reality. The term "Augmented Reality (AR)" denotes the superposition of additional virtual objects and supplementary information over real images. The joint project *Enhanced Reality (ER)*[1] aims at a generic AR-system. The ER-project is a cooperation of six different research groups of the Department of Computer Science at the University of Koblenz-Landau.

According to Ronald Azuma an AR-system combines real and virtual environments, where the real and virtual objects are registered in 3-D, and it provides interactivity in real time [Azu97]. Enhanced Reality extends Augmented Reality by requiring the virtual objects to be seamlessly embedded into the real world as photo-realistic objects according to the exact lighting conditions. Furthermore, additional information supplying value-added services may be displayed and interaction of the user may even be immersive.

The short-term goal of the ER-project is the exploration of *ER-fundamentals* using some specific research scenarios; the long-term goal is the development of a *component-based ER-framework* for the creation of ER-applications for arbitrary application areas.

ER-applications are developed as single-user applications for users who are moving in a real environment and are wearing some kind of visual output device like see-through glasses and some mobile end device. By these devices the user is able to see reality as it is, but he can also see the virtual objects and the additional information about some value-added service. Furthermore he might have additional devices whereby he can interact with the available virtual objects.

The development of a generic framework for ER-applications requires the definition of *generic components* which are customizable and composable to build concrete applications and it requires a *homogeneous data model* which supports all components equally well. The workgroup "Software Technology"[2] is responsible for this subproject. This report gives some preliminary results concerning the derivation of a component-based view of ER.

Aim of this work. There are several augmented reality frameworks like ARVIKA, AMIRE, DWARF, MORGAN, Studierstube and others which offer some support for the development of AR-applications. All of them ease the use of existing subsystems like AR-Toolkit, OpenGL and others and leverage the generation process for realistic systems by making efficient use of those subsystems. Consequently, they highly rely on them.

[1] http://er.uni-koblenz.de
[2] http://www.uni-koblenz.de/FB4/Institutes/IST/AGEbert

But, the reuse of existing subsystems in a framework heavily biases its structure, since properties of the reused parts have strong implications on the structure of the framework. Therefore, this paper approaches the problem from the opposite side: The structure of an ER-framework is derived by an extensive requirements analysis process which describes the necessary components solely driven by the demands of the given scenario and by the state of the art of AR-research.

Note by the authors. *Though the authors are aware of the fact that the efficient implementation of ER-systems will have to rely on existing subsystems in practice, any influence of existing packages on the requirements elicitation was avoided in order to derive a structure which primarily reflects the ER-domain, not the state of the practice.*

The framework structure described in this paper was derived during an elaborate process of identification, description, collection, comparison, categorization and unification of the most relevant entities in augmented reality system and their correlations. This work results in a first proposal for an overall structure of ER-applications, where the identified data and their correlations represent a basis for the *homogeneous data model* of an ER-framework and the identified activities represent *candidates for ER-framework components*. The claim of the paper is that a software structure derived this way is much more adequate for a generic framework. It should be easier to develop, easier to maintain, and much easier to use than a structure that is biased by the a priori necessity to make use of a given subsystem.

Component-based System. This work is inspired by the goal to construct an ER-framework which can be used to build concrete ER-applications by assembling appropriate customized components in some kind of wiring diagram.

This style of application development distinguishes to different roles: the application developer and the application user. The developer assembles an application and the user interacts with the applications.

Three different phases can be identified

- *assembly time*
  to actually build the application
  (this phase is done by the application developer)
- *initialisation time*
  to supply the application with the appropriate data
- *execution time*
  to use the running application

This paper focuses on the last phase since it determines the relevant components to be used in ER-applications. Chapter 2 gives a detailed description of a generic ER-application by identifying the most relevant activities and the data to be exchanged between them.

Furthermore, all data are identified which have to be supplied during assembly and/or initialisation for a successful application (figure 2.25).

Modelling. Since the derivation of a large set of widely applicable components is the main goal of this research, an *activity-oriented approach* has been chosen. UML-activity diagrams are used with a dataflow-oriented flavor. They describe the overall workflow in an ER-application by defining the relevant *activities* (services) and *objects* (devices and data). The activities identified in such a description are the candidates from which appropriate *components* in a component-based context will be derived in further work. Since this work is a first step towards defining an elaborated set of ER-components, a medium level of granularity is aimed at. The objects identified are the basis for a common *homogeneous data model*.

Note by the authors. *This paper is supposed to be a first step towards a generic component-based architecture. Since the description was compiled by software experts and not by ER-experts, this paper still has to be extensively discussed, criticised, amended and augmented before the next step - namely the formal definition of the components - can be done.*

All relevant entities (activities and data) are described in natural language and visualised via UML 2.0 [Obj06] activity and class diagrams in chapter 2; together these two kinds of explanation offer *a semi-formal specification*.

The description follows a dataflow-oriented style, since it aims at component-based systems, where the components are assembled by connecting their respective input/output data ports. The dataflow description is done using UML activity diagrams. The *activities* are candidates for components in a prospective system. The *data types* derived from their ports define the data to be exchanged. The description of the data types in a common class model (figure 2.26) gives hints which data will have to be integrated in a homogeneous data model for Enhanced Reality.

The activity diagrams are supposed to be consistent with the class diagrams in the sense that all ports and objects are typed by appropriate classes. The activity diagrams are hierarchical, i.e. each activity in the main diagram (figure 2.3) is refined by some other diagram in section 2.1 whose activities might be refined again.

The hierarchy of activity diagrams is balanced according to specialization. Here, balancing denotes the property that input/output ports of refining activity diagrams are consistent with the ports of the refined activity with respect to cardinality and specialization.

The data supplied to the system during system assembly and/or system initialisation are only included with those activities which make use of them.

Notational Conventions. To help in discussing this proposal with the domain experts, some minor additional conventions are being used in the UML activity and class diagrams.

Device objects and device classes are visualised by black rectangles using a white font. Within activity diagrams they are additionally marked by the stereotype “device”. *Data objects and data classes* are visualised by grey rectangles with a black border and a black font. Within activity diagrams they have the stereotype “datum”. These conventions make explicit that the domain of devices which might be supported by the ER-framework and the overall data model are separate concerns.

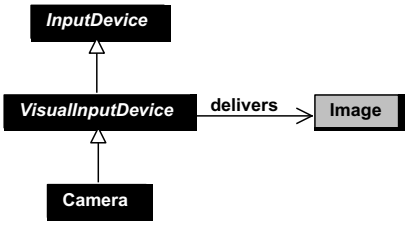

Figure 1.1: Conventions example for a class diagram

Activities are visualised by white rectangles with a black border and rounded corners. The arcs between activities and devices and/or data suggest a dataflow to or from the respective object. Thus, the input and output pins have also grey color. They are explicitly typed by a data type if they send or receive data to or from a device. If there is an asterisk next to a port in an activity diagram, zero or more objects can be used at that port in refinements of that particular activity.

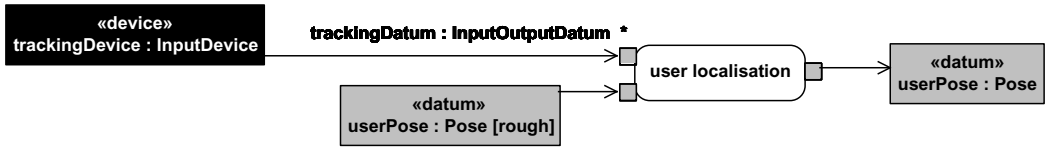

Figure 1.2: Conventions example for an activity diagram

The UML models were created using the *IBM Rational Software Modeler*[3]. Due to restrictions of this tool, objects are marked manually by an optional identifier and a colon in front their type only (not by underlining). Furthermore, since this tool is not able to mark the reading direction of associations, navigation arrows are used for denoting the direction of reading instead. Finally, since the tool does not check balancing conditions for the activities, their check was only done manually.

[3]http://www-306.ibm.com/software/awdtools/modeler/swmodeler/index.html

2 Enhanced Reality activities

This section describes the *ER-activities* which have to be performed before and during an ER-application. The identification of these activities in subsection 2.1 and subsection 2.2 is used to derive further information about the respective *ER-data* which have to be maintained to implement them. These data are structured into several packages and summarised in subsection 2.3.

Output devices. The primary objective of ER-systems is the superposition of visual information into the user's field of view. In the chosen scenario the user carries a visual output device which in some form provides this supplementary information. Examples are *optical see-through devices* which contain transparent displays. Further examples are *Video see-through devices* that are additionally combined with a camera which records the relevant part of reality. A small taxonomy of such devices is given in figure 2.1.

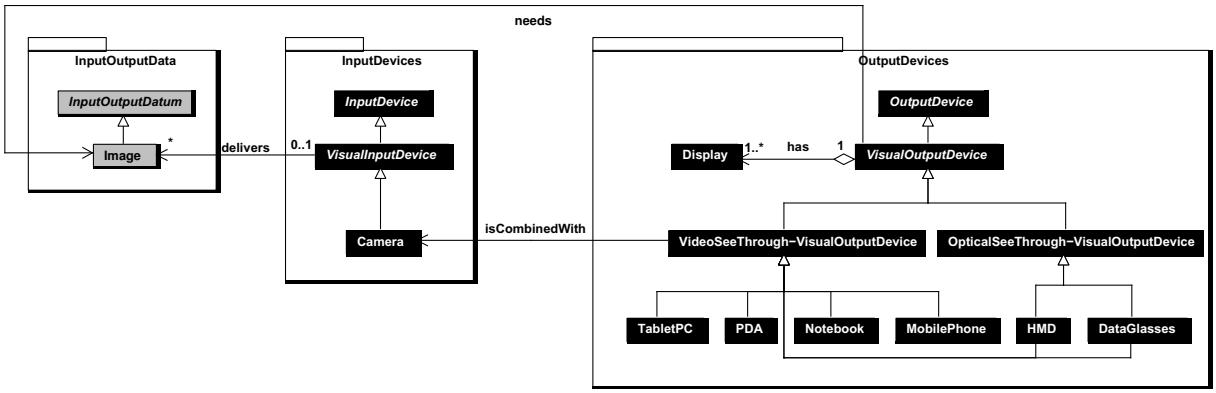

Figure 2.1: Output devices

Superimposable Information In the context of ER-systems two different kinds of information are to be distinguished which can be superimposed into a users field of view.

On the one hand there are *virtual objects*, i.e. images of non-existent real world objects or imaginable real world objects[1], which are to be superimposed at a certain pose. On the other hand there is *additional information* pertaining to real and/or virtual objects in a given scene. Such additional information could be *interaction facilities*, i.e. actions which a user may perform with

[1] e.g. a science fiction or fantasy object/character

a real or virtual object[2] or *value-added services*, i.e. services that make a user's situation more comfortable[3]. In these cases the user might even have to pay for such a service. (figure 2.2)

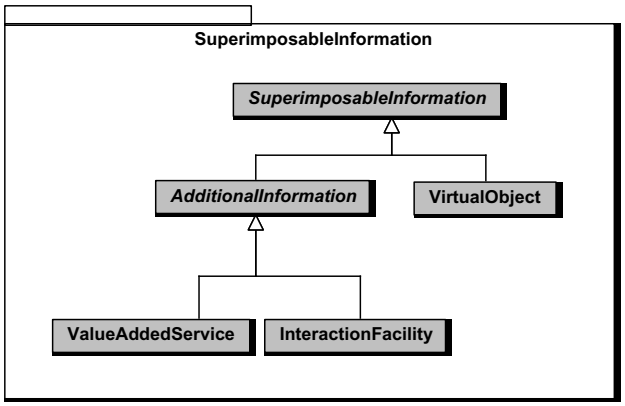

Figure 2.2: Superimposable information

Since the information to be superimposed should be adequate in a given situation, it has to be chosen intentionally depending on the context. This *context information* (user information and application information) as well as the *superimposable information* have to be *detected* and/or *processed* by some activities of the given ER-application.

Section overview. In the following, the main activities will be described in detail. At first there are activities being performed *during the creation* of an ER-application by the application developer (assembly). Then, there are activities which have to be performed *before the use* of the application by the user (initialisation), and at last there are activities being carried out automatically *during the use* of the application (execution).

At first, the activities executed during an ER-Application are discussed in detail (section 2.1). They are candidates for framework components in an ER-execution framework. Subsequently the activities during the creation of such an application and the activities at initialization time are discribed (section 2.2).

Note by the authors. *The detailedness of the different ER-activity descriptions varies because the activities contain different numbers of internal activities and the activities themselves are not equally well understood yet, as well. Furthermore some of them are well-established while others relate to the special goals of the ER-project. For some of the activities there exist a lot of different devices, and some of them need only one special device or a small group of devices.*

[2] e.g. a real door may be opened or any virtual object may be moved or transformed
[3] e.g. a movie program or the menu of a restaurant

2.1 Activities during the use of an Enhanced Reality application

There are several different activities which constitute an ER-application and which are executed at run time. The single activities are linked to each other via common data which implies a partial order as far as data dependency is concerned. Figure 2.3 gives a overview on the most important activities and their dependencies:

First of all one has to *localise the current user pose* (section 2.1.2) and the current *user input has to be processed* and potentially analyzed (section 2.1.1). Based on this information the system has to *select the output information* to be superimposed into the user's field of view (section 2.1.3). Furthermore the actual *lighting conditions have to be detected* (section 2.1.4) and the system has to *compute the relationship between the display and the user* (section 2.1.5). Using the latter information the *output image for the output device can be created* (section 2.1.6).

All these activities are explained in detail in the following subsections. For each of these activity a short natural language description and some additional explanation will be given. Then, to show how the respective activities might be refined in a potential application by more fine-grained activities, usually at least one such refinement is supplied, as well.

Note by the autors. *These refinements should be understood as examples only. A more systematic exploration of the set of possible refinements will be done later in the project, as soon as the overall structure proposed here will be adopted after an extensive discussion process.*

2.1.1 User input processing

Description. *User input processing* is the interpretation and (if necessary) analysis of the current *user input* (figure 2.4). For this purpose, the user carries one or more *user input devices* which deliver selective data and/or a permanent data stream containing the *user commands*.

There are several possibilities for a user to interact with a particular ER-system, depending on the various existing user input devices. First of all there are *haptical input devices*, like data gloves, mice, pointers, and the like. In most cases these devices provide one or more buttons to press or other mechanical gadgets that deliver events if the buttons are pressed or if the device itself is moved. These *mechanical events* can be directly interpreted as commands.

Moreover a user can also interact with a system using *visual* or *auditory input devices*, which deliver data that are continuous and have to be analyzed before they can be interpreted as commands. A visual user input device like a camera might deliver *images* that can contain *gestures* of the user; an auditory user input device like a microphone can deliver *sounds* that contain *utterances* of the user. The gestures as well as the utterance may then be interpreted as commands.

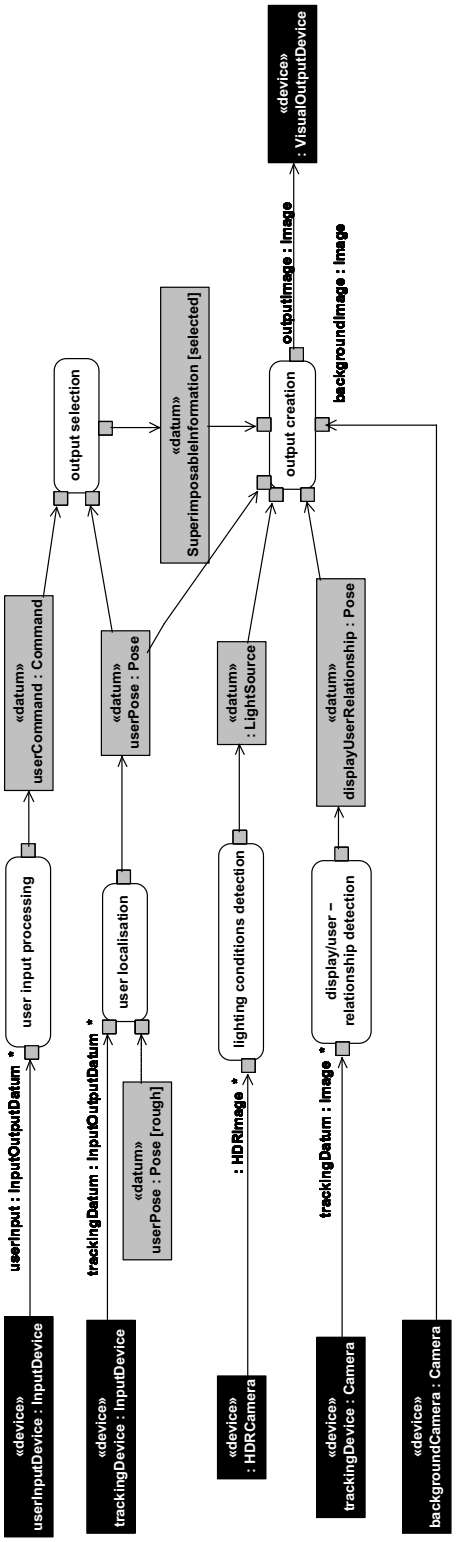

Figure 2.3: Activities during the use of an Enhanced Reality application

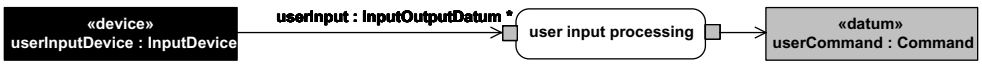

Figure 2.4: User input processing

Example refinement. In an ER-system an arbitrary number of user input devices can be combined. Then whereby the commands connected to the different kinds of user input have to be fused. Figure 2.5 visualises the user input processing using three different user input devices: one haptical, one visual and one auditory – but of course this is only one possible combination. Figure 2.6 gives a survey of the different kinds of user input and their corresponding user input devices as well as their relationships to each other.

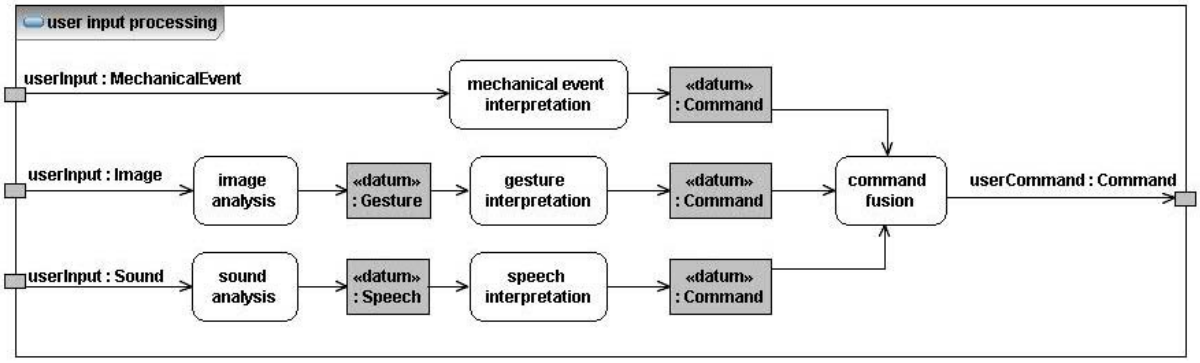

Figure 2.5: User input processing internal

Related Sub-Projects. In the scope of the ER-project the research group "Software Ergonomics"[4] does research in the field of user interaction of ER-systems in another subproject and wrote a state-of-the-art report about user input devices and user interaction metaphors for Augmented Reality [ST07].

[4] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGKrause

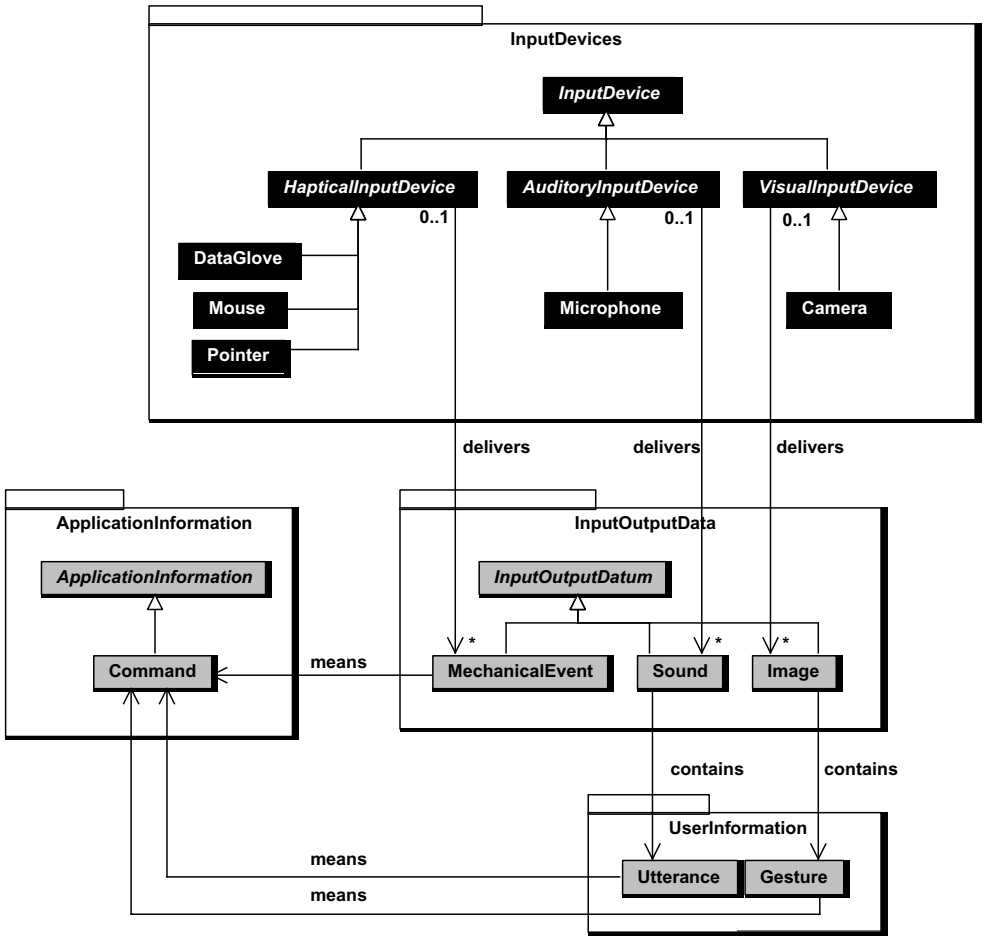

Figure 2.6: User input processing data and devices

2.1.2 User localisation

Description. *User localisation* is the detection of the current *pose of the user* with respect to the scene (figure 2.7). Therefore the user has to wear one or more *tracking devices* delivering a permanent data stream with which the user localisation can be done.

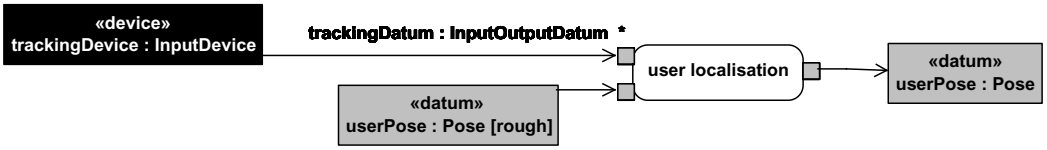

Figure 2.7: User localisation

The pose of a user consists of its *position* and *orientation* with respect to a *scene* which might be the *real world* or a specific *terrain* (which in turn is part of the real world). The position describes its spatial location and the orientation describes its "line of sight". Because a terrain has also a pose with respect to the real world, the real world pose and the terrain pose of a user can be translated into each other. Figure 2.8 demonstrates this connection.

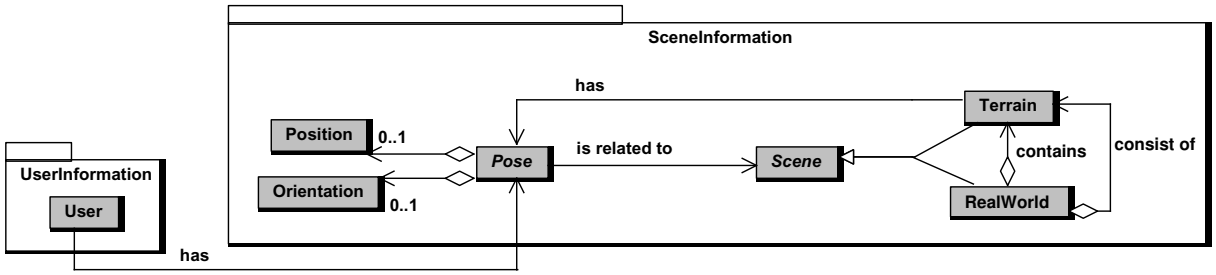

Figure 2.8: User pose

There are several different techniques for user localisation. Some of them deliver a pose, others can only detect a position or an orientation alone; therefore the term pose is often split into its two parts. Moreover the different techniques deliver pose, position and/or orientation with different degrees of accuracy; as a first coarse distinction one can distinguish between fine and rough user poses. Some of the techniques even need some initial pose information in order to refine it. And last but not least the different localisation techniques work under different conditions.

There are two different basic techniques for localisation, which also can be combined: *sensor based tracking* and *image based tracking*. In the following, these two kinds of localisation as well as their combination are discussed separately.

Example refinement 1: User localisation via sensor based tracking. Via sensor based tracking a *rough user position and/or orientation* can be detected with respect to the real world. Therefore the user has to wear one or more *sensors*, delivering a permanent sensor data stream. By processing these data the user position and/or orientation can be derived.

Various sensors deliver different results and work under special constraints. A WLAN[5]-receiver for example only functions in a prepared environment, where a wireless LAN is installed and enabled and where enough access points are available; it works indoor as well as outdoor. A GPS[6]-receiver however does not need any preparation – it functions all over the world as long as enough satellites are in sight; but it does not function inside of a building. Both receivers only deliver a position whereas a magnetic compass delivers an orientation, independent of its position – it functions indoor as well as outdoor but not close to magnetic materials. Therefore in most of the cases one has to combine different kinds of sensors in order to guarantee a user pose in every situation; the combination of two or more sensors is called *sensor fusion*.

Figure 2.9 visualises sensor based tracking with sensor fusion using three different kinds of sensors: one *sensor for position detection*, one *sensor for orientation detection* and one *sensor for pose detection*. Of course one could use only one of them or an arbitrary number of every sensor type. One should note, that even when using sensor fusion, the position and/or orientation of a user acquired by sensor based tracking is usually still rather rough.

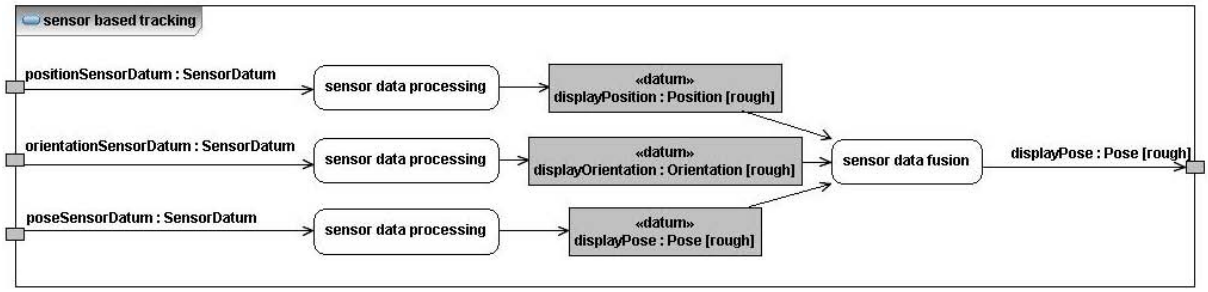

Figure 2.9: User localisation via sensor based tracking

Example refinement 2: User localisation via image based tracking. Via image based tracking a *fine user pose concerning a specific terrain* can be detected. Therefore the user has to wear one or more *cameras*[7], delivering a permanent image data stream. By processing these data the user pose can be derived.

There exist various image based tracking techniques. They all have in common that some supplementary information about the actual scene is required to do the user localisation. First of all one has to differentiate between *marker based image based tracking* ("marker based tracking" for short) and *marker less image based tracking* ("marker less tracking" for short).

When using marker based tracking, prior to the use of the ER-application one or more *markers*[8] have to be distributed in the environment and their poses have to be measured and kept

[5] Wireless Local Area Network
[6] Global Positioning System
[7] In the scope of this technical report we differentiate between cameras and other sensors, although cameras are also some kind of sensors.
[8] Markers are artificial Objects, in most cases pieces of paper containing a pattern, which offers a high recognition value compared to the environment.

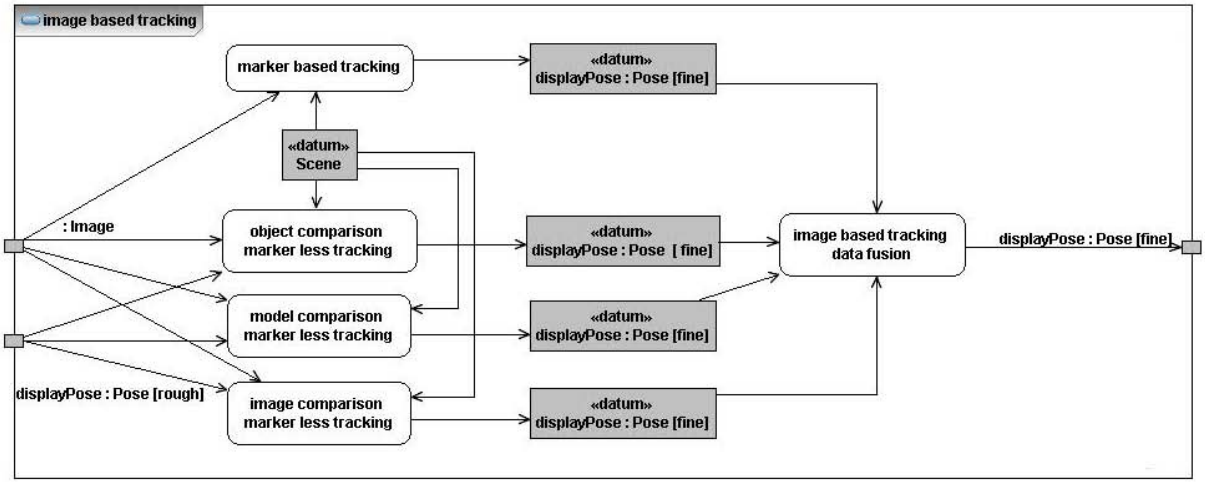

Figure 2.10: User localisation via image based tracking

(section 2.2). As soon as one of the markers is visible in a camera image the user pose can be derived.

When using marker less tracking, prior to the use of the ER-application some information about the *real objects* in the scene has to be collected (section 2.2). During the use of the application these real objects function as a kind of *natural markers*. As soon as one of the known real objects is visible in a camera image the user pose can be derived also.

The research area of marker less tracking is rather new. In the scope of the ER-project three methods for marker less tracking are being dealt with whose applicability shall be tested in the future:

- *model-comparison marker less tracking*
- *image-comparison marker less tracking*
- *object-semantic-comparison marker less tracking*

For the *model-comparison* method the camera images are segmented and the extracted *visual objects* are combined to a *three-dimensional model*. Then, this model is compared with the visual model of the complete actual scene, containing its *real objects* as well as their *poses* and *graphical descriptions*. Here, the complete model has to be created prior to the use of the ER-application (section 2.2) and both models have to be described in the same way. Initial information about a rough user pose can help to narrow the potential part of the scene.

The *image-comparison* method also needs the mentioned visual scene model. In this case, a rough user pose is required, because an image from this estimated pose is rendered from the visual scene model and afterwards this image is compared to the actual camera image.

For the *object-semantic-comparison* method the camera images are examined and some classifiable objects are searched. Therefore, information about the *semantical meaning* of the real objects has to be known so that a matching between found objects and real objects can be made. Here,

the semantical meaning is kept in a kind of *ontology*. Initial information about a rough pose can help again to reduce the search space.

Of course, all these different marker less image based tracking techniques as well as marker based tracking techniques can be combined to get the better result. Figure 2.10 visualises a combination of all mentioned image base tracking techniques.

Example refinement 3: User localisation as a combination of sensor and image based tracking. Naturally a combination between sensor and image based tracking can also be done. Sensor based tracking needs no initial information about a particular scene, but the results are rather rough. Image based tracking needs some information about a particular scene and in most cases initial information about a rough user pose are helpful. So it is a good idea to combine sensor and image based tracking sequentially. The rough pose delivered by the sensor based tracking functions is the used as an initial pose information for the image based tracking. Figure 2.11 visualises such a hybrid tracking technique.

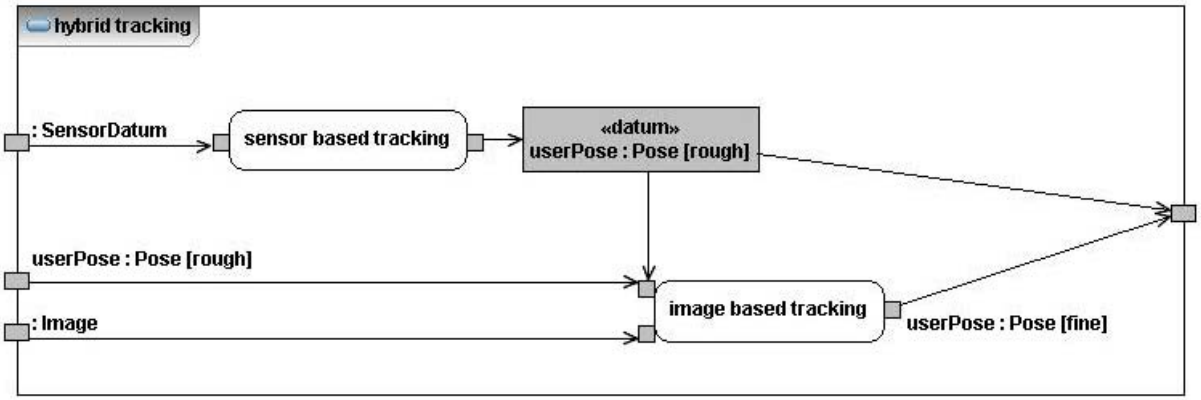

Figure 2.11: User localisation via hybrid tracking

Related Sub-Projects. In the scope of the ER-project the research group "Active Vision"[9] conducts a subproject on model-comparison and image-comparison image base tracking [PDKF07] and the laboratory "Image Recognition"[10] does research in the field of object-semantic-comparison image based tracking in another subproject.

[9] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGPaulus
[10] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGPriese

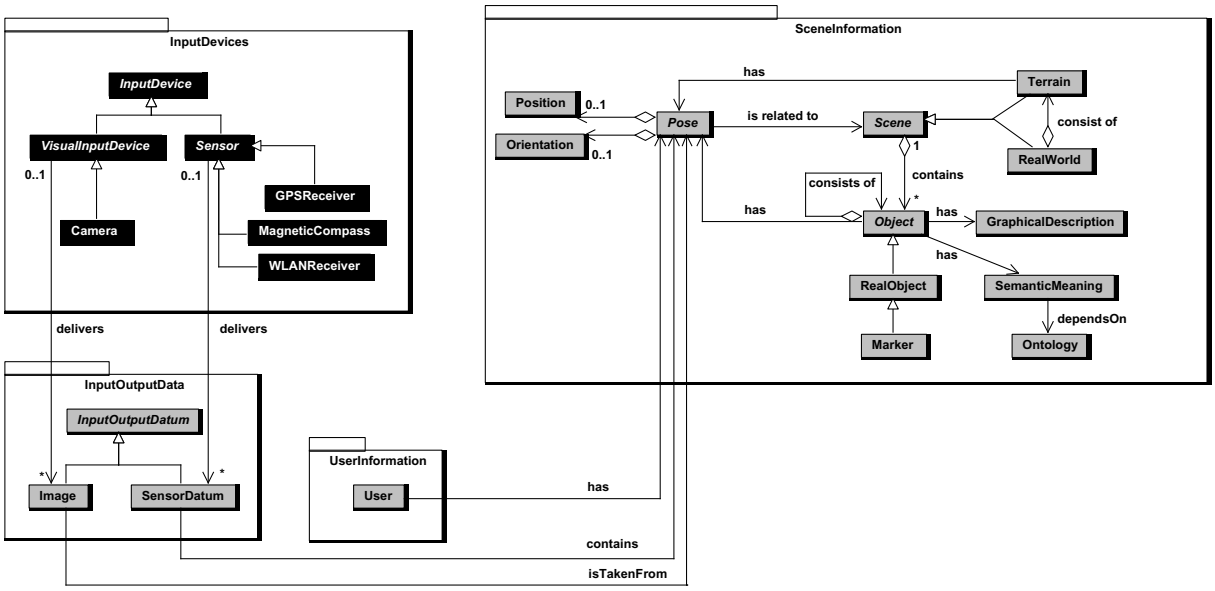

Figure 2.12: User localisation data and devices

2.1.3 Output selection

Description. Knowing the current *pose of the user* as well as the current *user commands*, the *information to be superimposed* into the users field of view can be selected (figure 2.13).

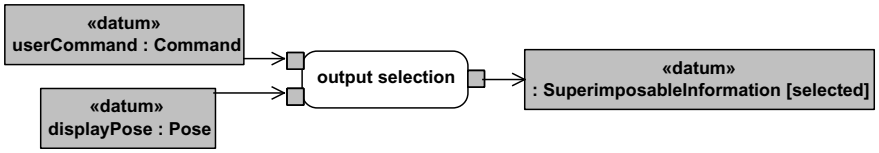

Figure 2.13: Output Selection

Example refinement. As mentioned before there are different kinds of *superimposable information* (figure 2.2). The *virtual objects* have got a pose concerning the scene and are embedded therein, and the *interaction facilities* constitute references to a real or virtual object within a scene. Using the current user pose all *possible superimposable information* with respect to his pose can be derived.

Concerning the decision *which* information shall be superimposed for a specific user at a particular point in time and in a given situation, there is some supplementary information needed. The possible superimposable information can be changed or restricted via the current *user commands* and/or some *context information*, namely *user information* or *application information*. Possible user information could be some *user preferences* the user has to make before the use of an application (section 2.2) or some *experience values* collected during other uses of the actual application or during other ER-applications (section 2.2).

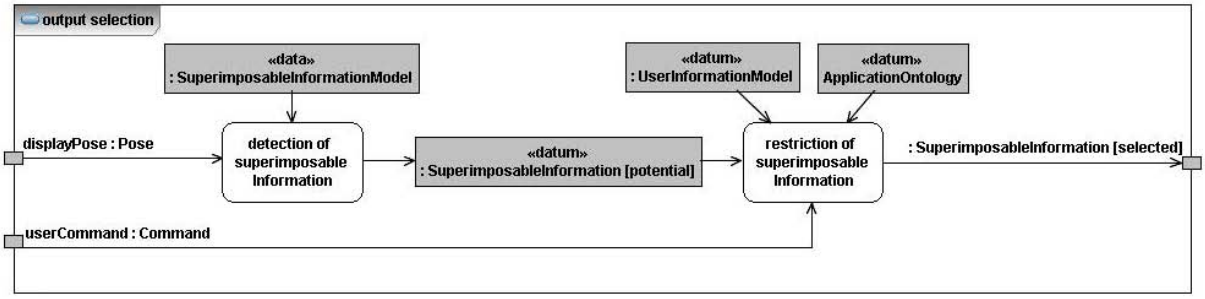

Figure 2.14: Output selection internal

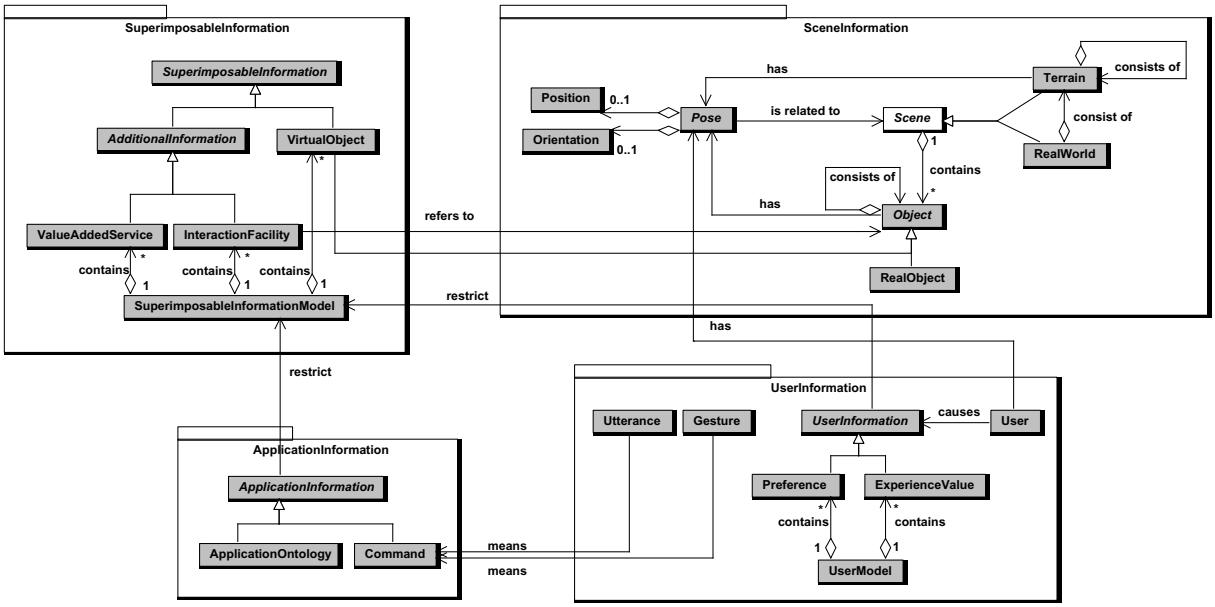

Figure 2.15: Output selection data

Feasible application information could be any *application ontology* containing knowledge about the application area, which has to be created before the use of an ER-application (section 2.2). So for example a user might have disabled a specific kind of value-added service in his user preferences or a particular kind of interaction facility might be given for the respective application area. The result of this step is the *selected superimposable information*. In contrast to the first step this second step is optional. Figure 2.14 visualises these internal output selection activities. Figure 2.15 visualises the data and devices required for this activity as well as their relationships.

Related Sub-Projects. None.

2.1.4 Lighting conditions detection

Description. *Lighting conditions detection* is the analysis of the environment in order to detect current *light sources* (figure 2.16). This information is important to render the output images as realistically as possible (section 2.1.6). Therefore the user carries one or more *HDR*[11] *cameras* delivering a permanent image stream from which the lighting conditions can be derived.

Figure 2.16: Lighting conditions detection

Example refinement. To detect the lighting conditions first of all *HDR images* have to be analyzed to find the potential light sources. Afterwards the three-dimensional poses of these located light sources have to be derived with respect to the scene. This can be done using an image stream, images of a HDR stereo-camera or rather images of two HDR cameras. (figure 2.17)

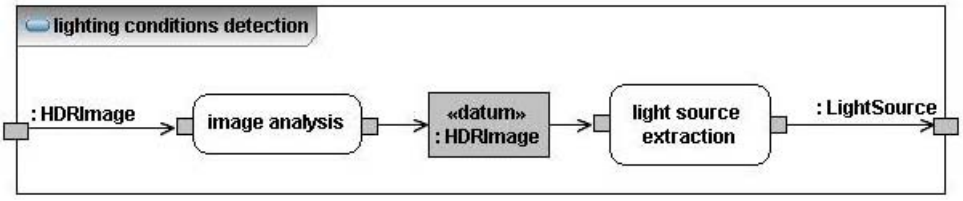

Figure 2.17: Lighting conditions detection internal

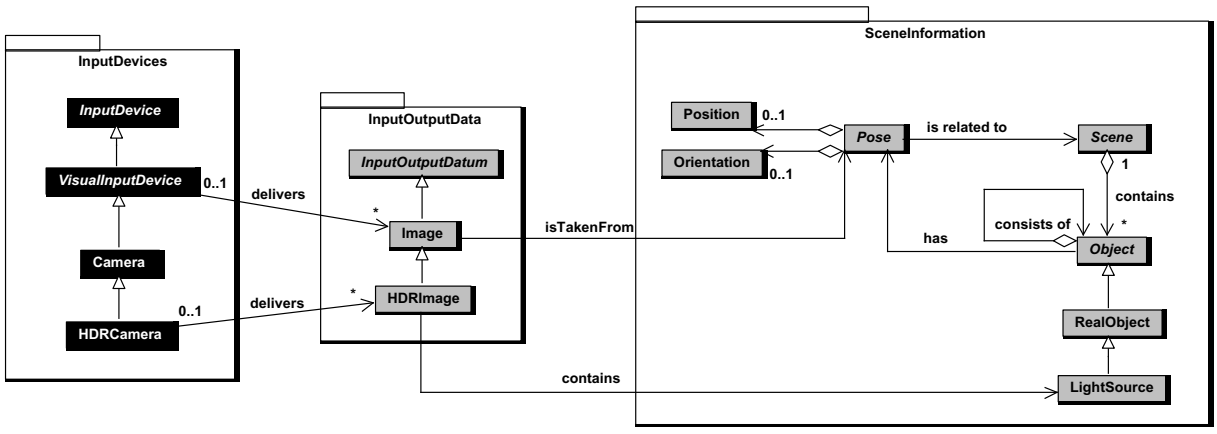

Figure 2.18: Lighting conditions detection data and devices

[11] High Dynamic Range

Related Sub-Projects. In the scope of the ER-project the research group "Computer Graphics"[12] does research in the field of light source detection using a HDR stereo-camera in another subproject [KSv+06].

2.1.5 Display/user relationship detection

Description. The actual *relationship* between the *output device* or more precisely its *display(s)*[13] and the *user* (figure 2.21) is required to render the output images as realistically as possible (section 2.1.6). Therefore there should be a camera attached to the output device tracking the user from the output devices display(s) point of view.

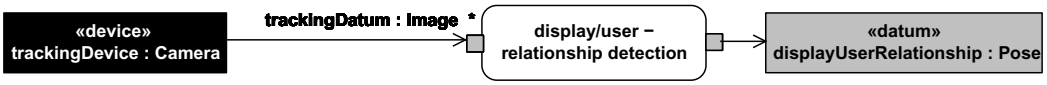

Figure 2.19: Display/user relationship detection

Example refinement. For the tracking of the user all *image based tracking methods* described above (section 2.1.2) can be used: marker based tracking as well as the marker less tracking methods (figure 2.20). Figure 2.21 visualises the data and devices required for this activity as well as their relationships. Pure sensor based tracking however is usually much too rough for the detection of the display/user-relationship.

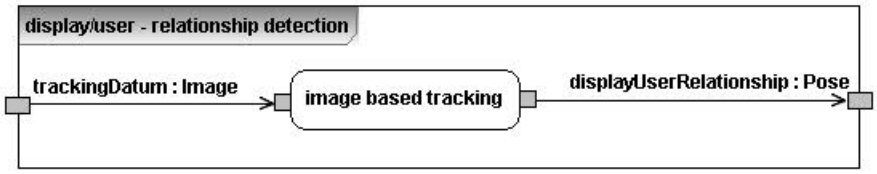

Figure 2.20: Display/user relationship detection internal

Related Sub-Projects. In the scope of the ER-project the laboratory "Image Recognition"[14] works in the field of the automatic detection of the display/user relationship for optical see-through output devices using the user's eyes as a kind of natural marker combined with some infrared techniques [PSL07].

[12] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGMueller
[13] There could be visual output devices containing more than one display, data glasses for example.
[14] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGPriese

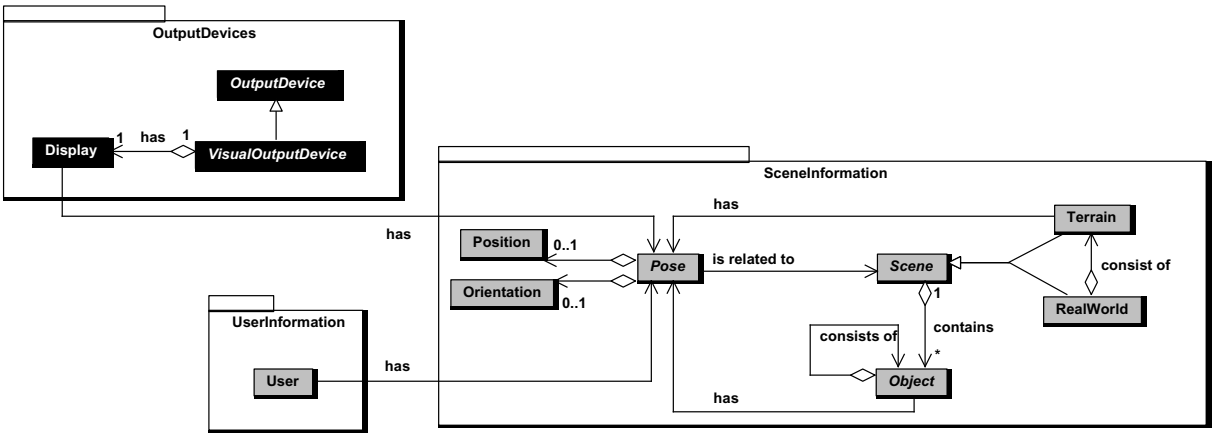

Figure 2.21: Display/user relationship detection data and devices

2.1.6 Output Creation

Description. Knowing the *selected superimposable information*, the current *display pose*, the current *light sources* and the current *display user relationship*, the information to be superimposed into the users field of view can be created (figure 2.22).

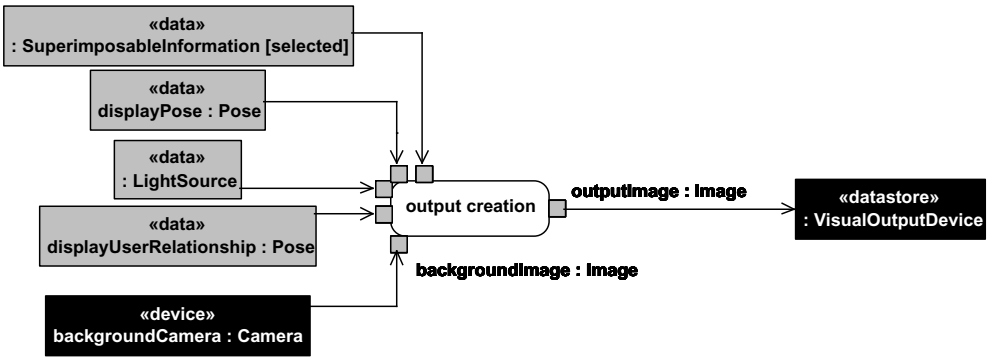

Figure 2.22: Output creation

Example refinement. In the scope of the output creation there coud be three different image parts which have to be created. First of all there is the *image of the visual objects*; moreover there is the *image containing the additional information*; an last but not least if a video see-through device is used there is the *background image*. These internal activities concerning the output creation are explained in the following.

For the *rendering of the virtual objects image part* different kinds of information is required. First of all one has to know the *selected virtual objects* as well as their poses. These virtual objects should not be silhouetted against the reality, because the user should not be able to decide, whether an object in his field of view is real or virtual. Thus we need images of the objects,

which are as lifelike as possible. This means the image has to be suited to its pose as well as to the lighting conditions of its environment as well as possible.

Normally the only way to create such an image is building a three-dimensional model of the object, which can be rendered from any pose and with any lighting conditions. Therefore not only one image of an object to be superimposed is required, but also its three-dimensional graphical description. These objects have to be created prior to the use of the ER-application (section 2.2). Knowing the current *user pose* and the current *display user relationship* as well as the current *light sources*, an image for the user reflecting this situation can be created.

For the *visualisation of the additional information* the selected additional information, i.e. the feasible interaction facilities and the achievable value-added services, as well as the actual lighting conditions are required.

In contrast to the virtual information these data shall be silhouetted against the reality to attract the user's attention, and - also in contrast to the virtual objects - there is no predefined representation for additional information. Normally they could be visualised for example via texts with specific symbols, but this depends on the interaction metaphor of the specific ER-application as well as on other context information. Using a typical visualization style for the application - which also had to be created before the use of an ER-application (section 2.2) - the additional information is visualised also respecting the actual lighting conditions.

After the two image parts were created they are combined to a *common output image*, whereby the additional information superimposes the virtual information. If a video see-through device is used the background image is also combined with the output image, whereby the additional information as well as the virtual information superimpose the background image. Afterwards this output image is adapted for the specific output device and finally delivered to this device. Figure 2.23 visualises these internal output selection activities.

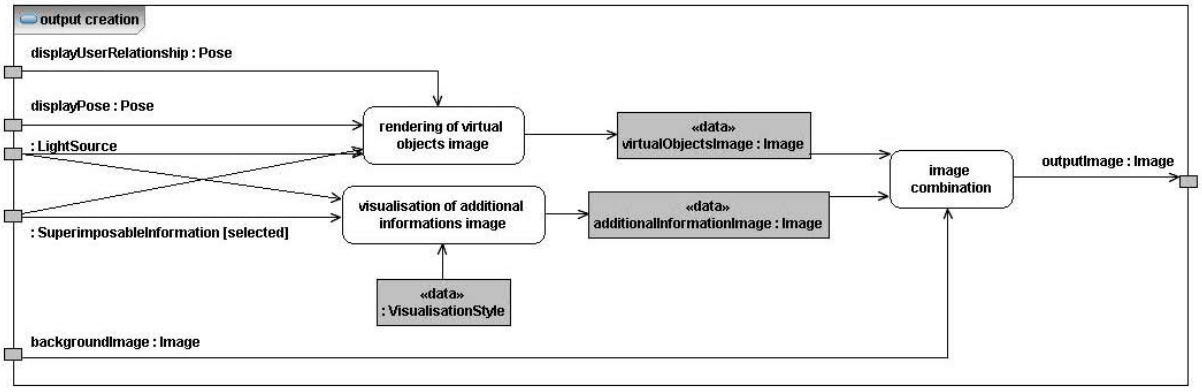

Figure 2.23: Output creation internal

Related Sub-Projects. In the scope of the ER-project the research group "Computer Graphics"[15] does research in the field of interactive rendering of virtual objects with respect to a given

[15] http://www.uni-koblenz.de/FB4/Institutes/ICV/AGMueller

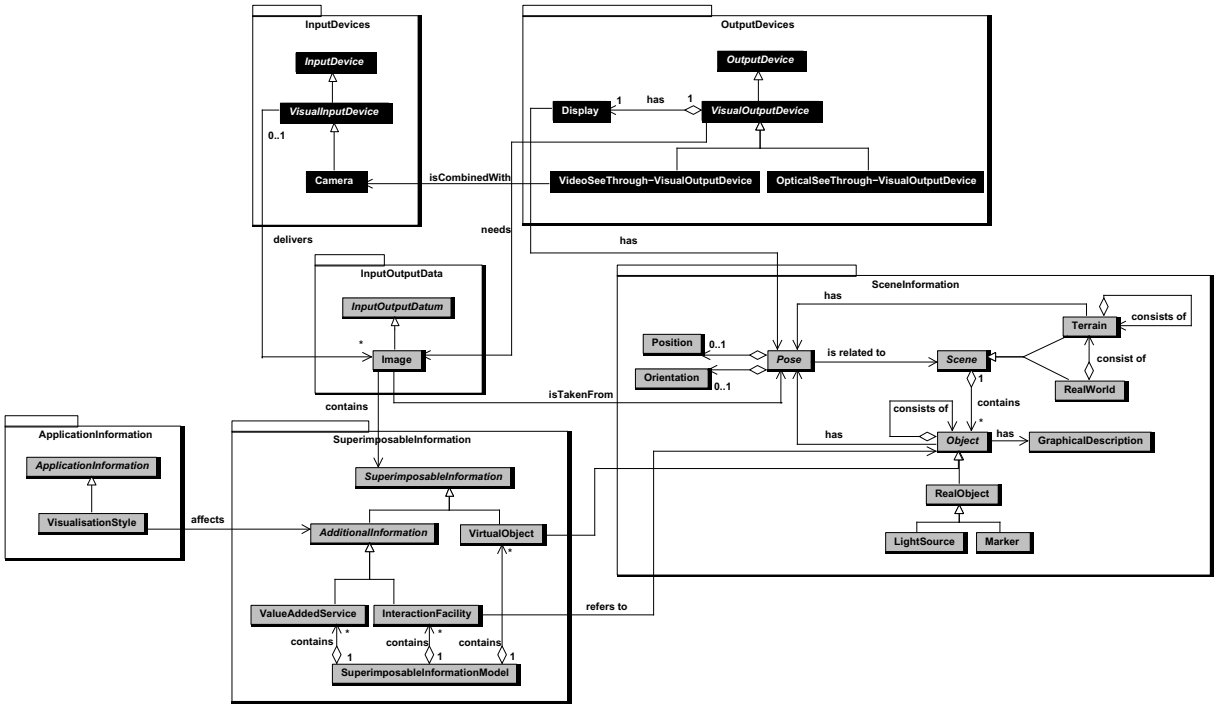

Figure 2.24: Output creation data and devices

scene in another subproject [RG06, KSv+06].

2.2 Activities prior to the use of an Enhanced Reality application

Activities at Initialisation Time. In the preceding section some activities have been mentioned that have to be performed *prior to the use* of an ER-application at initialization time.

An ER-application has to be prepared prior to its use with respect to its *user* on the one side and with respect to a a particular set of input and output *devices* on the other side:

- When preparing the system for the user some user *preferences* – entered by the user – and some user *experience values* – stored during some former use of the application – can be combined to a basic *user model*.
- When preparing the system for a set of devices, the respective input- and output devices are to be identified, configured, and calibrated for the user and the scene as well as harmonised with each other.

If marker based tracking is used, the scene has also to be prepared by distributing the markers at the scene poses that had been chosen during the creation of the ER-application.

Activities at Assembly Time. There are also some activities that have to be performed *during the creation* of an ER-application.

One task during the creation of an ER-application is the creation of the *virtual objects* as well as the detection of the *interaction facilities* and the *value-added services* and their composition to a superimposable information model.

The most important activity is probably the detection of the *real objects* concerning a real scene and the composition of a scene model out of these objects. Depending on the used localisation technique(s), the graphical description of a real object and/or his semantic meaning have to be detected and modelled too, and if the semantic meaning is needed, an adequate ontology for all the semantic meanings is required. The created virtual objects have also be connected to the scene model although they are resident in the superimposable information model.

A specialisation of the real object detection is the detection of the ideal *marker poses*, if marker based tracking is used. The markers or more precise their selected marker poses have also be inserted into the scene model.

Finally, the application information has to be prepared. This means on the one hand the creation of an *application ontology* and the extension of the ontology belonging to the scene information with this application ontology. And on the other hand there is the creation of one or more *visualisation styles* for the ER-application.

Figure 2.25 shows a kind of context diagram, visualising the Enhanced Reality application as a kind of activity as well as all data that have to be detected or created prior the use of an Enhanced Reality application.

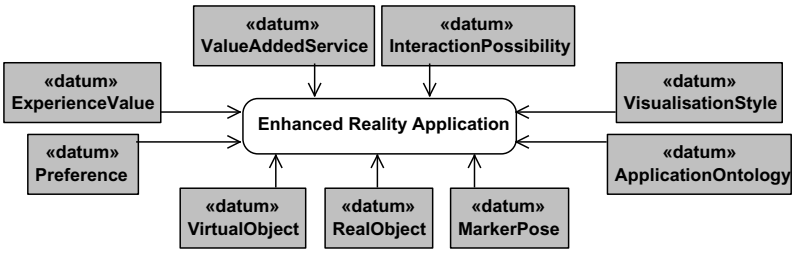

Figure 2.25: Activities prior the use of an Enhanced Reality application

2.3 Enhanced Reality data

The activities described above make use of data which have already been introduced in the various class diagrams in section 2.1 and section 2.2. While the activity definitions are the precursors of the *components* to be defined more formally in a component-based system, the data and their interrelations are early descriptions of the *homogenous data model* that the underlying environment should provide to the components.

In this paper, the data are structured using *packages*. This is done to separate the different concerns in ER-systems appropriately. This section focusses on the data and summarises the main packages used. All packages are displayed together in figure 2.26 in their context.

Note by the authors. *None of the packages is assumed to be final, yet. On the contrary, the details of the packages are subject to discussion and depend on the activities that use them. Here the packages are elaborated only to that extent that is necessary for defining the main activities of an ER-system.*

Devices package The package *Devices* contains the classes of all devices, that have been mentioned in the description of one or more of the activities in the section 2.1. *Input Devices* and *Output Devices* are described in different sub-packages by two small taxonomies, though in practice many physical devices are combinations of input and output parts.

All *input devices* are subsumed in a superclass *Input Device* which is specialised into four different types *Haptical Input Devices* (e.g. *Data Glove*, *Mouse* and *Pointer*), *Auditory Input Device*(e.g. *Microphone*), *Visual Input Device* (e.g. *Camera*), and *Sensor* (e.g. *GPS Receiver*, *Magnetical Compass* and *WLAN Receiver*).

Since the scenarios discussed in the ER-project do only focus on visual output, the superclass *Output Device* is only specialised to the one subclass *Visual Output Device* which stands for device classes like *Tablet PC*, *PDA*, *Notebook*, *Mobile Phone*, *HMD* and *Data Glasses*. This taxonomy will be extended later as needed. All visual output devices have one or more *Displays*. Those are categorised into *Optical see-through Visual Output Devices* and *Video see-through Visual Output Devices*.

Input and output data package All data that connect a running ER-system with its environment are packaged together as *Input and Output Data*. This package is important for the embedding of an ER-system at runtime.

The package contains all events and data that are delivered to the system by input devices and those that are needed by output-devices to be displayed to the user, where the latter are restricted to images as the only visual output up to now (cf. 2.3). The main categories of input and output data are described by the classes *Mechanical Event*, *Sound*, *Image* and *Sensor Datum*.

Superimposable information package All information that is additive to real image is modelled in the package *Superimposable Information*. This information is used by the activity *output creation* to superimpose generated image parts on the given current image of reality which extend the reality perceived by the user.

According to the scenarios tackled by the ER-project the cited classes *Virtual Object* and *Value-added Service* and *Interaction Facility* are distinguished. As mentioned before, virtual objects are images of real world objects or imaginable real world objects, which can be superimposed at a pose where they do not exist. Interaction possibilities are actions which a user can perform with a real or virtual object, value-added services are services that make a user's situation more comfortable.

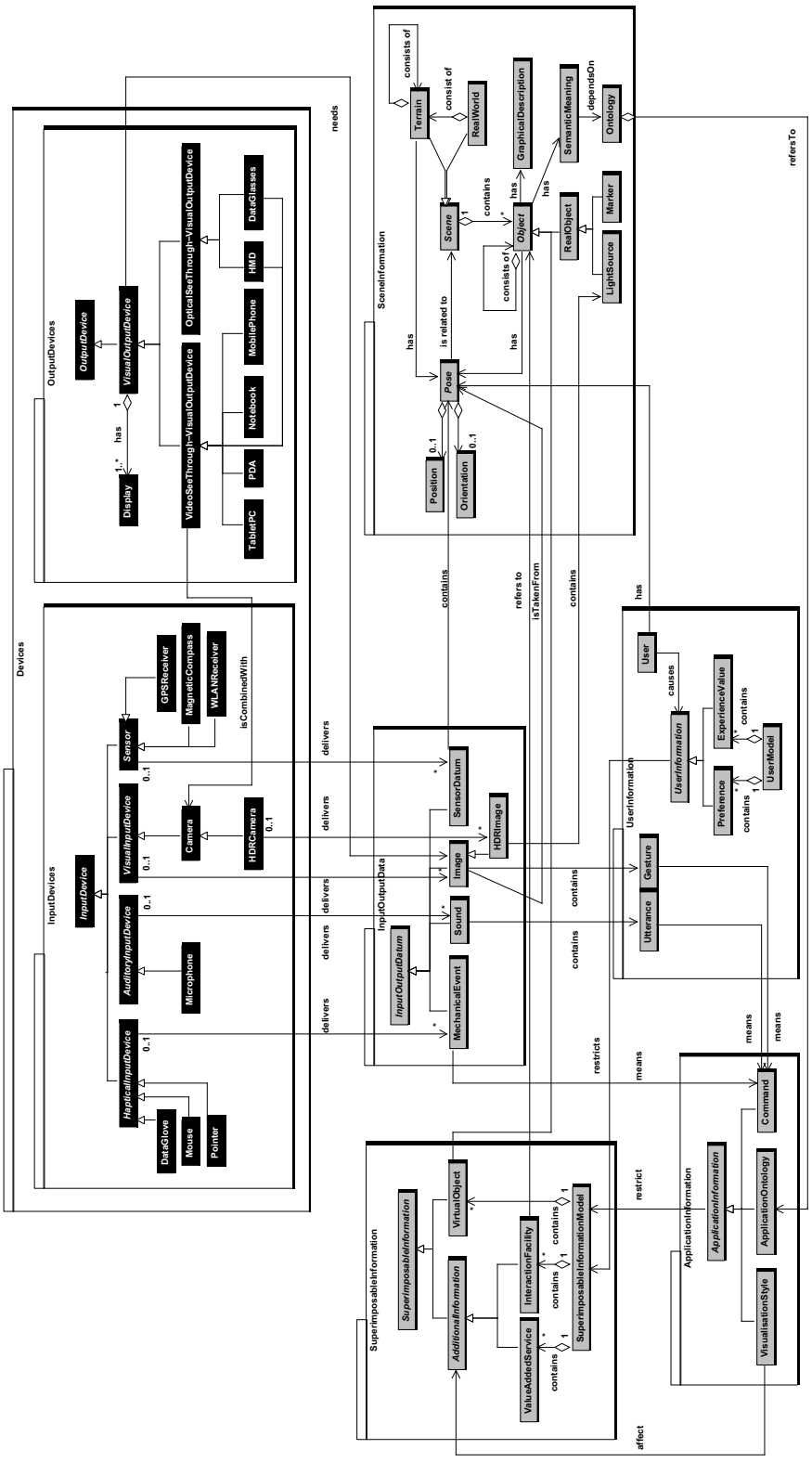

Figure 2.26: Survey of Enhanced Reality Data

User information package During the execution of an ER-application all data pertaining to the user are to be kept in a rudimentary user model. These data are modelled by the classes in the package *User Information*. It contains all data that are connected to the user.

At present the information connected to the user is restricted to the classes *Preference* and *Experience Value*. Preferences are settings done by the user himself before or during the usage of the application. Experience values are data that are derived by the system itself in order to be reused in later sessions. The user model also contains Gesture or utterance that a user might give to the running system via some input device and that express commands (application information package).

Application information package Application information is all kind of coded knowledge about the ER-application itself including the application domain. It is packaged in *Application Information*.

An example for the information needed about the application itself might be the *Visualisation Style* used to superimpose additional information into the generated image (cf. *SuperimposableInformation*). The knowledge about the domain might be encapsulated in some *Application Ontology*. The application information also contains the *Commands* that a user might give to the running system via some input device. Commands can be expressed by *Gestures* or *Utterances* (user information package).

Scene information package The package *SceneInformation* is the central part of the data model. Here all runtime information concerning the current scene is kept. This information has to be detailed enough for all activities (subactivities) of an ER-application and it has to contain the relevant information for all generative and all analytical activities of the areas of computer graphics and image processing.

The main class is the class *Scene* which contains *Objects*. These objects might be real or virtual and have to be modelled by a *Graphical Description* and may be attributed by some *Semantical Meaning* from some *Ontology*.

The scene information also contains all kinds of poses. Here it is assumed that the *Real World* is partitioned into different *Terrains* which contain the *Objects*. The placement and orientation of objects and terrains is described by their (absolute or relative) *Poses*.

Note by the authors. *It is a research project for itself to detail this data model and - even more - to derive efficient data representations that support real-time execution of all relevant activities on this bases. The structure given here does not yet fulfil these requirements. It is only intended to sketch the main contents roughly.*

3 Conclusion

Enhanced Reality systems are expected to get more and more accepted in future applications since the necessary hardware (like see-through glass, portable cameras and location devices) will become lighter and cheaper soon.

In this paper, the authors presented a *decomposition* of ER-systems into their main activities and subactivities which are supposed to be candidates for components and subcomponents in such a system. This decomposition was done in a dataflow-like manner and was purely driven by the logical structure of the tasks which are identifiable in existing ER-solutions. It was intentionally avoided to let this structure be driven by the need to use existing APIs or subsystems. Following good practice in software development all efficiency aspects have been postponed.

All *activities* and their interconnection was described semiformally by hierarchical activity diagrams. Thereby the *data* needed or supplied by the activities could be identified in details, as well. Thus, the decomposition also delivered an elaborated conceptual description of the data inherent in ER-systems.

Both parts, the activities and the data, will be discussed, adapted and refined in the ER-project to achieve a wide-accepted general model. This model will be used as a basis for formally specifying the components and the data model of generic ER-systems. After deciding on the technical basis of the intended component-like system, a first experimental implementation will be tried. Then, efficiency aspect in choosing appropriate data structures and meeting real time conditions will be tackled.

Bibliography

[Azu97] Ronald T. Azuma. A survey of Augmented Reality. *PRESENCE: Teleoperators and Virtual Environments*, 6(4):355–385, 8 1997.

[KSv+06] Matthias Korn, Maik Stange, Andreas von Arb, Lisa Blum, Michael Kreil, Kathrin-Jennifer Kunze, Jens Anhenn, Timo Wallrath, and Thorsten Grosch. Interactive Augmentation of Live Images using HDR Stereo Camera. In Müller and Zachmann [MZ06].

[MZ06] Stefan Müller and Gabriel Zachmann, editors. *3. Workshop Virtuelle und Erweiterte Realität der GI-Fachgruppe VR/AR*, number ISBN 3-8322-5474-9. Shaker Verlag, 2006.

[Obj06] Object Management Group. *Unified Modeling Language: Superstructure*. Object Management Group, 2006.

[PDKF07] Dietrich Paulus, Frank Deinzer, Alexander Kubias, and Tobias Feldman. Image Based Registration, Tracking and Reconstruction. Technical report, Universität Koblenz-Landau, Institut für Computervisualistik, Arbeitsgruppe Aktives Sehen, 2007. to be published.

[PSL07] Lutz Priese, Frank Schmitt, and Paul Lemke. Automatische See-Through Kalibrierung. Arbeitsberichte aus dem Fachbereich Informatik 07/2007, Universität Koblenz-Landau, Institut für Computervisualistik, Labor Bilderkennen, 01 2007.

[RG06] Tobias Ritschel and Thorsten Grosch. On-line Estimation of Diffuse Materials. In Müller and Zachmann [MZ06].

[ST07] Philipp Schaer and Marco Thum. State-of-the-Art: Interaktion in Erweiterten Realitäten. Arbeitsberichte aus dem Fachbereich Informatik 10/2007, Universität Koblenz-Landau, Institut für Computervisualistik, Arbeitsgruppe Softwareergonomie und Information Retrieval, 01 2007.

Bisher erschienen

Arbeitsberichte aus dem Fachbereich Informatik
(http://www.uni-koblenz.de/fb4/publikationen/arbeitsberichte)

Jürgen Ebert, Kerstin Falkowski: A First Proposal for an Overall Structure of an Enhanced Reality Framework, Arbeitsberichte aus dem Fachbereich Informatik, 8/2007

Lutz Priese, Frank Schmitt, Paul Lemke: Automatische See-Through Kalibrierung, Arbeitsberichte aus dem Fachbereich Informatik, 7/2007

Rüdiger Grimm, Robert Krimmer, Nils Meißner, Kai Reinhard, Melanie Volkamer, Marcel Weinand, Jörg Helbach: Security Requirements for Non-political Internet Voting, Arbeitsberichte aus dem Fachbereich Informatik, 6/2007

Daniel Bildhauer, Volker Riediger, Hannes Schwarz, Sascha Strauß, „grUML – Eine UML-basierte Modellierungssprache für T-Graphen“, Arbeitsberichte aus dem Fachbereich Informatik, 5/2007

Richard Arndt, Steffen Staab, Raphaël Troncy, Lynda Hardman: Adding Formal Semantics to MPEG-7: Designing a Well Founded Multimedia Ontology for the Web, Arbeitsberichte aus dem Fachbereich Informatik, 4/2007

Simon Schenk, Steffen Staab: Networked RDF Graphs, Arbeitsberichte aus dem Fachbereich Informatik, 3/2007

Rüdiger Grimm, Helge Hundacker, Anastasia Meletiadou: Anwendungsbeispiele für Kryptographie, Arbeitsberichte aus dem Fachbereich Informatik, 2/2007

Anastasia Meletiadou, J. Felix Hampe: Begriffsbestimmung und erwartete Trends im IT-Risk-Management, Arbeitsberichte aus dem Fachbereich Informatik, 1/2007

„Gelbe Reihe“
(http://www.uni-koblenz.de/fb4/publikationen/gelbereihe)

Lutz Priese: Some Examples of Semi-rational and Non-semi-rational DAG Languages. Extended Version, Fachberichte Informatik 3-2006

Kurt Lautenbach, Stephan Philippi, and Alexander Pinl: Bayesian Networks and Petri Nets, Fachberichte Informatik 2-2006

Rainer Gimnich and Andreas Winter: Workshop Software-Reengineering und Services, Fachberichte Informatik 1-2006

Kurt Lautenbach and Alexander Pinl: Probability Propagation in Petri Nets, Fachberichte Informatik 16-2005

Rainer Gimnich, Uwe Kaiser, and Andreas Winter: 2. Workshop "Reengineering Prozesse" – Software Migration, Fachberichte Informatik 15-2005

Jan Murray, Frieder Stolzenburg, and Toshiaki Arai: Hybrid State Machines with Timed Synchronization for Multi-Robot System Specification, Fachberichte Informatik 14-2005

Reinhold Letz: FTP 2005 – Fifth International Workshop on First-Order Theorem Proving, Fachberichte Informatik 13-2005

Bernhard Beckert: TABLEAUX 2005 – Position Papers and Tutorial Descriptions, Fachberichte Informatik 12-2005

Dietrich Paulus and Detlev Droege: Mixed-reality as a challenge to image understanding and artificial intelligence, Fachberichte Informatik 11-2005

Jürgen Sauer: 19. Workshop Planen, Scheduling und Konfigurieren / Entwerfen, Fachberichte Informatik 10-2005

Pascal Hitzler, Carsten Lutz, and Gerd Stumme: Foundational Aspects of Ontologies, Fachberichte Informatik 9-2005

Joachim Baumeister and Dietmar Seipel: Knowledge Engineering and Software Engineering, Fachberichte Informatik 8-2005

Benno Stein and Sven Meier zu Eißen: Proceedings of the Second International Workshop on Text-Based Information Retrieval, Fachberichte Informatik 7-2005

Andreas Winter and Jürgen Ebert: Metamodel-driven Service Interoperability, Fachberichte Informatik 6-2005

Joschka Boedecker, Norbert Michael Mayer, Masaki Ogino, Rodrigo da Silva Guerra, Masaaki Kikuchi, and Minoru Asada: Getting closer: How Simulation and Humanoid League can benefit from each other, Fachberichte Informatik 5-2005

Torsten Gipp and Jürgen Ebert: Web Engineering does profit from a Functional Approach, Fachberichte Informatik 4-2005

Oliver Obst, Anita Maas, and Joschka Boedecker: HTN Planning for Flexible Coordination Of Multiagent Team Behavior, Fachberichte Informatik 3-2005

Andreas von Hessling, Thomas Kleemann, and Alex Sinner: Semantic User Profiles and their Applications in a Mobile Environment, Fachberichte Informatik 2-2005

Heni Ben Amor and Achim Rettinger: Intelligent Exploration for Genetic Algorithms – Using Self-Organizing Maps in Evolutionary Computation, Fachberichte Informatik 1-2005