In: Narayanan, S. (Ed.). 12th IFAC, IFIP, IFORS, IEA Symposium on Analysis, Design and Evaluation of Human-Machine-Systems. Las Vegas : International Federation of Automatic Control. pp. 483-490

# The Usability Engineering Repository UsER for the Development of Task- and Event-based Human-Machine-Interfaces

**Michael Herczeg. Marc Kammler. Tilo Mentler. Amelie Roenspieß**

*Institute for Multimedia and Interactive Systems (IMIS), University of Luebeck, Luebeck, Germany (e-mail: herczeg@imis.uni-luebeck.de).*

**Abstract:** The Usability Engineering Repository (UsER) is a flexible development environment to support collaborative analysis, design and evaluation of interactive human-machine systems. For this purpose, UsER provides several modules, which cover different aspects and methods of the development of human-machine interfaces. UsER supports the contextualized development of user interfaces in a broad range of application areas like office systems as well as safety-critical control systems by providing general as well as domain specific analysis, design and evaluation modules. These modules may be applied as needed and their contents will be cross-referenced through linked entities and hypermedia relationships. This semantic network created through analysis, design and evaluation can be mapped into linear document structures for formal development purposes, especially for project deliverables and contracts. UsER can be integrated with other development environments through a standardized requirements interface as well as a standardized process model interface.

*Keywords:* Usability Engineering, Cognitive Systems Engineering, Human Machine Systems

## 1. INTRODUCTION

Today, a variety of development process models for software engineering exist, which are suitable in various application domains. However, the analysis, design and evaluation of user-oriented requirements are still not sufficiently taken into account in most development projects, because either the process model does not support the appropriate consideration of user requirements or the software development environment lacks integrated tooling for user-centered development. The first issue has been addressed to some extent by introducing new development processes, like agile and user-centered software development. In order to tackle the second issue, the development environment needs to support methods for collaborative and iterative usability engineering. The Usability Engineering Repository (UsER), which we have been developing through the last three years together with industrial partners, is supposed to fill this gap while being applicable independently of the software development and production environments in use. With the UsER system the processes and the tooling in place may be extended and enriched for user-centered development.

As there are different questions, challenges and needs in different application domains, the analysis, design and evaluation methods applied will differ. While standard office applications are basically task-based, safety-critical control systems have to be task- as well as event-based. In UsER we support a broad range of applications by providing a selection of modules as instantiations of development methods that can be applied for specific projects. The contents created with these modules will be interlinked by hypermedia structures and by cross-referencing semantic entity-relationships.

## 2. THE PROCESS AND APPROACH

### 2.1 The Process

The basic process for a UsER-based development follows the now widely accepted principles of user-centered development according to ISO 9241-210. The process phases are supported by different modules of the UsER environment (Fig. 1).

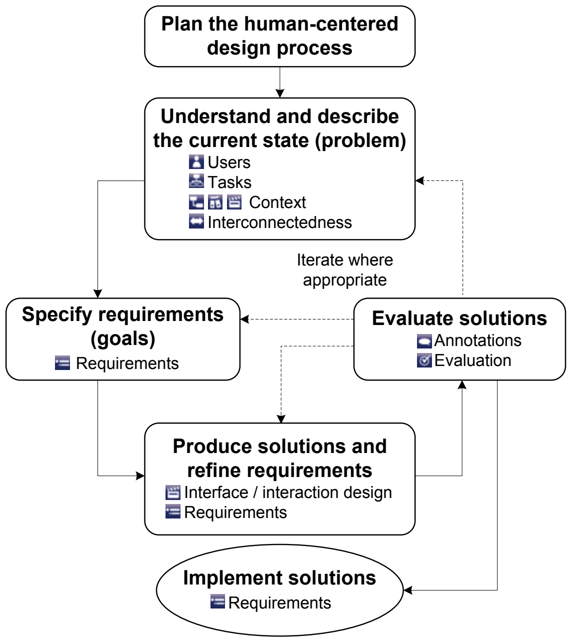
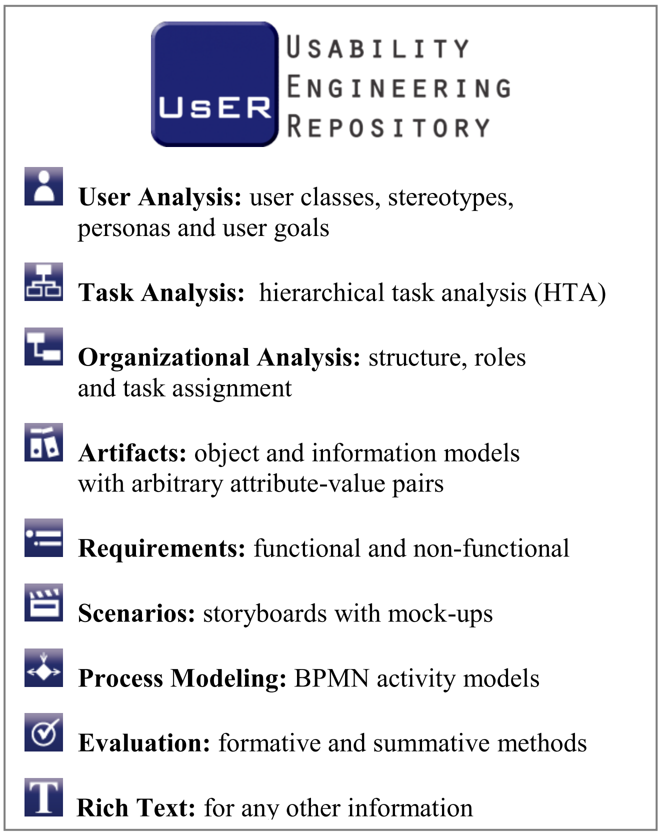

Plan the human-centered design process
Understand and describe the current state (problem)
Users
Tasks
Context
Interconnectedness
Iterate where appropriate
Specify requirements (goals)
Requirements
Evaluate solutions
Annotations
Evaluation
Produce solutions and refine requirements
Interface / interaction design
Requirements
Implement solutions
Requirements

Fig. 1. The Basic Process Model for UsER

### 2.2 The Modules

The flexibility of UsER stems from its variety of method modules (Fig. 2) and their ability to be combined and interrelated. The modules might be chosen for specific projects, depending on the goals and requirements. Their contents, which are represented by conceptual entities or hypermedia content, can be interrelated via hypermedia links and semantic entity-relationship associations.


UsER USABILITY ENGINEERING REPOSITORY

**User Analysis:** user classes, stereotypes, personas and user goals

**Task Analysis:** hierarchical task analysis (HTA)

**Organizational Analysis:** structure, roles and task assignment

**Artifacts:** object and information models with arbitrary attribute-value pairs

**Requirements:** functional and non-functional

**Scenarios:** storyboards with mock-ups

**Process Modeling:** BPMN activity models

**Evaluation:** formative and summative methods

**Rich Text:** for any other information

Fig. 2. The modules of UsER

The modules listed in Figure 2 are just the current repertoire of methods in UsER. New modules for additional or new methods may be integrated easily into the open framework as required. A few other method modules are already under development in an ongoing process. The extensibility of the environment is a basic feature possible by its open modular Java-, GWT- and MySQL-based architecture.

## 3. GENERAL PRINCIPLES

UsER provides a platform to describe, annotate, collect, aggregate, cross-reference and version user- and application-centered requirements for interactive systems, organized as projects in an integrated interactive environment. As a foundation, several general principles have been applied: creating entities, iteratively specifying their attributes, semantic linking of entities as well as annotating and communicating entities between participants in the development process, including customer and end users.

### 3.1 Modeling and Specifying Entities

UsER supports the classical phases of system development except implementation: analysis, design, evaluation and reflective usage and feedback during the operational phase. In respect to the activities in these phases, UsER allows to define, structure and describe the entities of a work system. In the analysis phase these are mainly organizational entities and structures, roles and competences, tasks and application objects (artifacts), users and their capabilities as well as problem scenarios combining these entities into stories and contexts of standard, non-standard and abnormal usage. In the design phase there are interaction scenarios and mock-ups with features, services or application functions derived from the tasks. During the implementation and evaluation phases these analysis and design entities may be annotated, discussed, refined and finalized. The evaluation phase will check through task-based test cases how effective, efficient, satisfactory and safe the tasks will be performed by users in certain roles and predefined work contexts. In the operational phase user feedback, typically through evaluations, tickets or incident reports may be fed back into the development loops.

### 3.2 Collecting and Linking Entities

Within UsER the analysis and design entities defined will be structured and cross-referenced. Every entity may be semantically linked to any other entity within the same project or referring to certain global entities (templates) defined through other projects. This helps to interrelate analytic findings, design concepts and structures within one project or across projects. The result will be a structured collection of information and requirements about users, organizations, roles, tasks, physical and informational artifacts and objects, processes, scenarios as well as any informal rich-text information and external assets. These meshed entities may be sequentialized in documentations to create and print reports for official development documents or contracts like system and software specifications on different levels and in different phases of the development process.

### 3.3 Annotating and Communicating Entities

In order to communicate problems and concepts directly within the relevant development context where they arise, annotations, documents and other materials (assets) may be attached to any entity. Textual annotations and assets like graphics, screen shots, hotline tickets, incident reports etc. may be addressed and sent to one or more recipients. The project repository will grow by new information sources, findings and decisions, which can be stored and saved in subsequent versions of the repository. Communication between users, customers, analysts, developers, external experts, even across phases of the development process, will be controlled by a workflow engine within UsER. The system has been developed partially in a bootstrapping process by applying its own methods onto itself as an interactive software product.

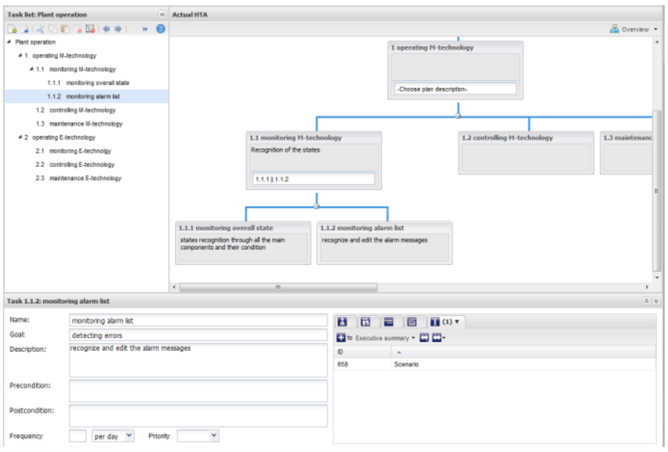
## 4. ORGANISATIONS, TASKS AND ROLES

A canonical start for the analysis of a process control system is the analysis and definition of tasks, roles and organizational structures. UsER provides a module for organizational structures, roles and staffing as well as a module for task analysis.

### *4.1 Task Analysis*

Generally the development of operational systems starts with the analysis of tasks. There is a long history of methods for task analysis. The method of hierarchical task analysis (HTA) is rewarding and therefore widely used in many domains (Annett & Duncan, 1967; Shepherd, 1998, Stanton, 2006). While many design questions may be addressed by a task analysis, new questions often arise and lead the development into a new direction. An HTA for an analyzed work system will usually be modified and optimized using the claims derived from problem scenarios, resulting in an HTA for the target work system to be implemented.

Tasks may be defined in UsER as a hierarchical graph or a textual structure (Fig. 3). Each node in the HTA may be selected and described with pre- and post-conditions for the task to be performed as well as further attributes like duration, frequency, or criticality when needed. Additionally, each task node can be linked to a role entity within the UsER project for task allocation.


Task list: Plant operation
Actual HTA
Overview
Plant operation
1 operating M-technology
1.1 monitoring M-technology
1.1.1 monitoring overall state
1.1.2 monitoring alarm list
1.2 controlling M-technology
1.3 maintenance M-technology
2 operating E-technology
2.1 monitoring E-technology
2.2 controlling E-technology
2.3 maintenance E-technology
1 operating M-technology
-Choose plan description-
1.1 monitoring M-technology
Recognition of the states
1.1.1 ‖ 1.1.2
1.2 controlling M-technology
1.3 maintenanc
1.1.1 monitoring overall state
states recognition through all the main components and their condition
1.1.2 monitoring alarm list
recognize and edit the alarm messages
to Executive summary

Task 1.1.2: monitoring alarm list

| | |
|---|---|
| Name: | monitoring alarm list |
| Goal: | detecting errors |
| Description: | recognize and edit the alarm messages |
| Precondition: | |
| Postcondition: | |
| Frequency: | per day |
| Priority: | |

| ID | |
|---|---|
| 658 | Scenario |

Fig. 3. HTA in UsER for a plant control system (detail)

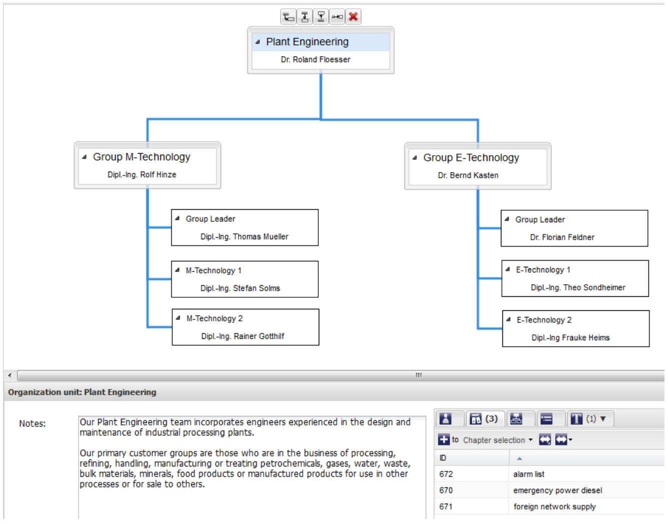
As task structures are directly related to organizational structures and roles, it is quite typical to perform an organizational analysis in parallel.

Usually roles in task analysis will be for human operators but machine roles may be defined in the same way to define task allocation between humans and machine in the sense of flexible automation and supervisory control systems.

### *4.2 Organizational Analysis*

A control system will always be embedded into an organization. The operators and supervisors will be part of an organizational structure and will hold positions with certain roles. This structure can be depicted in organizational charts and linked to tasks and work object entities (section 5.1).

UsER allows depicting and refining a typical form of org-chart and allows the positions being associated to roles, while the roles are associated with tasks and work objects (Fig. 4). The organizational structure serves as the place where positions, roles and tasks are clarified or synthesized for new operational and organizational patterns. Other than typical tools for organizational analysis, UsER binds the entities for positions, roles and tasks together and supports analysis and optimization of the operational and work breakdown structure as well as the consequences of automation.



Organization unit: Plant Engineering

Notes: Our Plant Engineering team incorporates engineers experienced in the design and maintenance of industrial processing plants.

Our primary customer groups are those who are in the business of processing, refining, handling, manufacturing or treating petrochemicals, gases, water, waste, bulk materials, minerals, food products or manufactured products for use in other processes or for sale to others.

| ID | |
|---|---|
| 672 | alarm list |
| 670 | emergency power diesel |
| 671 | foreign network supply |

Fig. 4. Organizational chart for a plant control system (detail)

### *4.3 Process Modeling*

Describing work structures for a control system will always imply to describe work processes. UsER provides a module for process modeling (Fig. 5) based on a subset of BPMN 2.0, the international standard for business process modeling. Activities of operators may be defined in different kinds of process charts as needed.

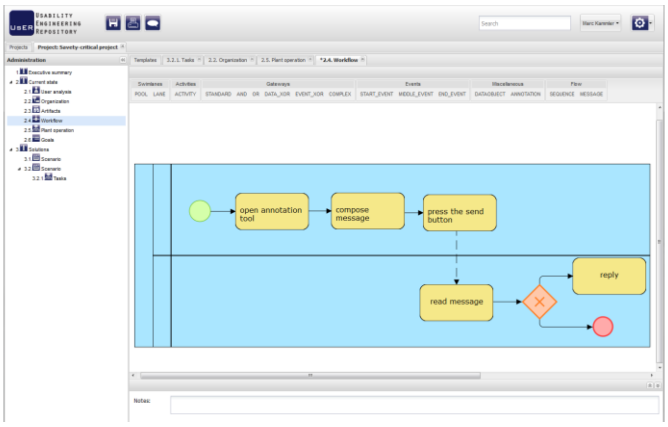


Fig. 5. Process chart in UsER (detail) based on BPMN 2.0

## 5. WORK AND CONTROL OBJECTS

Process control in an abstract sense means the supervision and control through an information model representing the work objects, which have to be observed and manipulated in their states whenever necessary. Work objects from the application domain itself will usually not be controlled and manipulated directly and physically. The process control system serves as intermediate information and interaction layer providing control objects that will be perceived and manipulated as substitutes.

To design a process control system means to define work objects and perhaps several levels of abstraction with control objects replacing the domain objects in the operational tasks. Rasmussen proposes two dimensions for this structure: the abstraction dimension, from the purpose down to the physical level, and the decomposition hierarchy of domain or informational objects from the whole to the parts (Fig. 6; cf. Rasmussen, 1984, 1985; Rasmussen et al., 1994; Vicente, 1999).

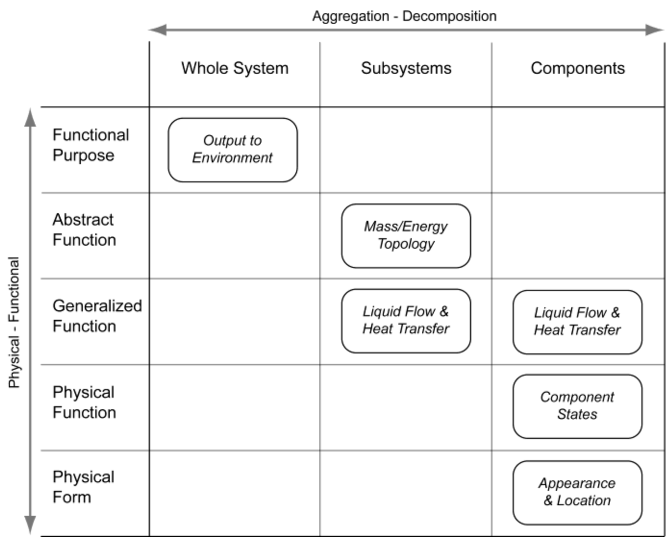


Fig. 6. Functional abstraction and decomposition of work objects (cf. Rasmussen, 1985)

The UsER system provides a module that allows the definition of artifacts, which may be the representatives of these structures of work and control objects. This module provides a foundation for the information model of a control system.

### *5.1 Work Objects*

Thinking in work objects, means thinking in the application domain itself. For example for a plant control system the domain objects will be pumps, tanks, valves or even the plant as a whole. It might be helpful to start an analysis by defining the essential work objects from an operational view with their basic attributes to be controlled. From there it is possible to come up with the initial operational task structures and operator roles. UsER allows to associate artifacts with tasks and roles to clarify responsibilities and activities.

### *5.2 Control Objects*

As already mentioned, there will be an abstraction of work objects into control or informational objects. These may be positioned into several layers representing the different layers of an operational functional abstraction (cf. Fig. 6). These different informational layers may serve as operational layers themselves, as the operational tasks and procedures will be structured according to the informational abstractions and vice versa.

## 6. USERS, SCENARIOS AND CONTEXTS

Usability engineering has often been called to be user- as well as application-centered. This basically means that the target users and their work situations will be the focus points for the system design. This can be done systematically by user modeling and the description of scenarios and work contexts for the operators and other roles in the context of a control system.

UsER provides a flexible user modeling tool as well as a rich-text hypermedia scenario editor to elaborate on users, their characteristics and activities.

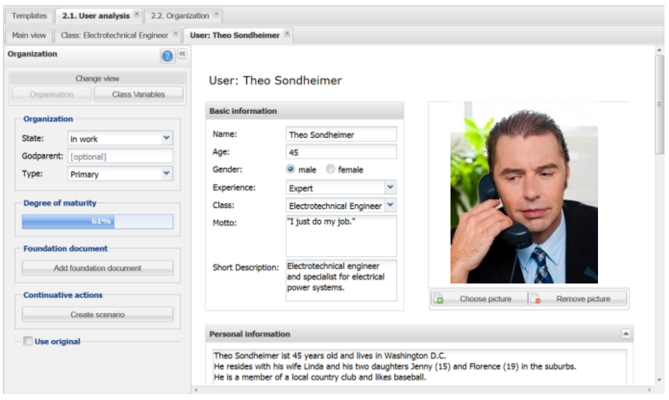
### *6.1 User Modeling*

User modeling means to analyze the target users, their capabilities as well as their limits, their expectations and their mental models (Johnson-Laird, 1983; Carroll & Olsen, 1988). User modeling has been done in many different ways. There are methods for informal descriptions of users, like in the Persona Model (Cooper, 1999), as well as more formal descriptions like GOMS (John, 1995) or other cognitive engineering methods.

UsER provides a user class modeling framework, where users may be described in respect to some basic and abstract properties as well as more specific and illustrative by describing stereotypes or even personas (Fig. 7).

User classes are abstract descriptions like for example operators in general. A stereotype might for example be *"the experienced electrical engineer being in service for more than 15 years for a production company"*. A persona might for example be *"Theo Sondheimer, a 45 year old electrotechnical engineer working for the company Quick Pac as an expert for electrical power systems."*

UsER allows creating more or less abstract user models directly out of field research by defining surveys and variables through the evaluation module (see section 9).



Organization

State: in work
Godparent: [optional]
Type: Primary

User: Theo Sondheimer

Basic information

Name: Theo Sondheimer
Age: 45
Gender: [✓ male] female
Experience: Expert
Class: Electrotechnical Engineer
Motto: "I just do my job."
Short Description: Electrotechnical engineer and specialist for electrical power systems.

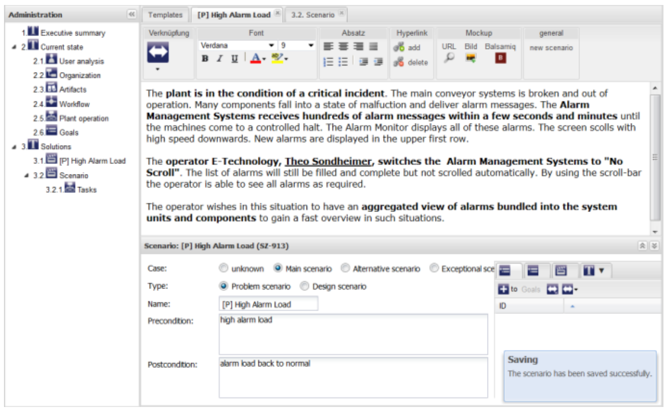
Personal information

Theo Sondheimer ist 45 years old and lives in Washington D.C.
He resides with his wife Linda and his two daughters Jenny (15) and Florence (19) in the suburbs.
He is a member of a local country club and likes baseball.

Fig. 7. Persona model in UsER
(Image courtesy of stockimages / FreeDigitalPhotos.net)

## *6.2 Scenarios and Contexts*

It is widely considered important to involve the stakeholders, especially the operators and operations experts into the design process. For this, all contributors in the development process need a common language and common reference points in the problem domain. Rich text *problem scenarios* (Fig. 8) can serve this purpose right at the start of the process, because they show and verbalize use cases in a readable, understandable and problem-oriented way, other than abstract depictions like UML Use Case Diagrams. On one hand, the development team needs to understand the conditions and requirements at the user's workplace. On the other hand, users should get an idea of how the system will look and feel when it will be available. The difficulty to derive *interaction scenarios* or *design scenarios* from problem scenarios can be mitigated by *claims*. Claims are used to describe features of the current work and usage situation which have relevant positive or negative consequences (Rosson & Carroll, 2002). During the design phase, claims are helpful to point out potential positive and negative aspects of design decisions and clarify problems from the original starting points. Scenarios can be enriched by mock-ups of the human-machine interfaces.



The **plant is in the condition of a critical incident**. The main conveyor systems is broken and out of operation. Many components fall into a state of malfuction and deliver alarm messages. The **Alarm Management Systems receives hundreds of alarm messages within a few seconds and minutes** until the machines come to a controlled halt. The Alarm Monitor displays all of these alarms. The screen scolls with high speed downwards. New alarms are displayed in the upper first row.

The **operator E-Technology, Theo Sondheimer, switches the Alarm Management Systems to "No Scroll"**. The list of alarms will still be filled and complete but not scrolled automatically. By using the scroll-bar the operator is able to see all alarms as required.

The operator wishes in this situation to have an **aggregated view of alarms bundled into the system units and components** to gain a fast overview in such situations.

Scenario: [P] High Alarm Load (SZ-913)

Case: unknown [✓ Main scenario] Alternative scenario Exceptional sce
Type: [✓ Problem scenario] Design scenario
Name: [P] High Alarm Load
Precondition: high alarm load
Postcondition: alarm load back to normal

Saving
The scenario has been saved successfully.

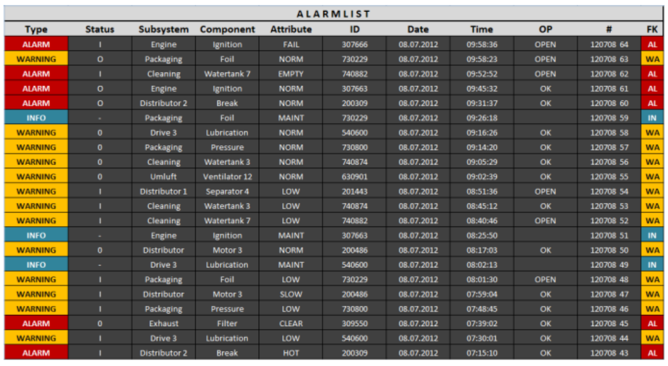
Fig. 8. Problem scenario in UsER for an alarm list for a plant control system

## 7. DESIGN SKETCHES AND MOCKUPS

After having done a good part of the problem analysis and modeling, the human-machine system has to be designed. This basically means workspace design along with some details for displays and controls. The result may be more detailed sketches or specifications of an operator's workspace like a cockpit or an industrial control room.

### *7.1 Mock-ups and Screenshots*

*Low-fidelity (lo-fi) mock-ups* are a method to design human-machine interfaces. The basic idea of mock-ups is the construction of an interaction concept by focusing on the basic functions of an interactive system. The attribute "lo-fi" refers to the low level of graphical and functional detail and the lack of actual programmed behaviour. Main points of interest are rather simple shapes, layouts and visual codings of interaction elements than the precise later appearance. Therefore it is sometimes been recommended to make the mock-ups even look like sketches or drawings. This ensures that all stakeholders involved will be aware of working on a changeable sketch without caring about details too soon. Further details will be treated in later design phases. Lo-fi mock-ups can be interpreted as low-fidelity prototypes, which will become more and more detailed during the process. Lo-fi mock-ups incorporate a variety of previously collected information from the scenarios and models. The stakeholders, who are involved in the design process, create lo-fi mock-ups by using the design scenarios, drawings, charts, screenshots, experiences with existing tools, individual preferences and various other inputs. The operators shall be involved in this iterative lo-fi mock-up creation process. The advantage of lo-fi mock-ups is that all previously determined requirements concerning the users are transformed into something "graspable" and iteratively improved by all of the participating stakeholders and system designers.



ALARMLIST

| Type | Status | Subsystem | Component | Attribute | ID | Date | Time | OP | # | FK |
|---|---|---|---|---|---|---|---|---|---|---|
| ALARM | I | Engine | Ignition | FAIL | 307666 | 08.07.2012 | 09:58:36 | OPEN | 120708 64 | AL |
| WARNING | O | Packaging | Foil | NORM | 730229 | 08.07.2012 | 09:58:23 | OPEN | 120708 63 | WA |
| ALARM | I | Cleaning | Watertank 7 | EMPTY | 740882 | 08.07.2012 | 09:52:52 | OPEN | 120708 62 | AL |
| ALARM | O | Engine | Ignition | NORM | 307663 | 08.07.2012 | 09:45:32 | OK | 120708 61 | AL |
| ALARM | O | Distributor 2 | Break | NORM | 200309 | 08.07.2012 | 09:31:37 | OK | 120708 60 | AL |
| INFO | - | Packaging | Foil | MAINT | 730229 | 08.07.2012 | 09:26:18 | | 120708 59 | IN |
| WARNING | 0 | Drive 3 | Lubrication | NORM | 540600 | 08.07.2012 | 09:16:26 | OK | 120708 58 | WA |
| WARNING | 0 | Packaging | Pressure | NORM | 730800 | 08.07.2012 | 09:14:20 | OK | 120708 57 | WA |
| WARNING | 0 | Cleaning | Watertank 3 | NORM | 740874 | 08.07.2012 | 09:05:29 | OK | 120708 56 | WA |
| WARNING | 0 | Umluft | Ventilator 12 | NORM | 630901 | 08.07.2012 | 09:02:39 | OK | 120708 55 | WA |
| WARNING | I | Distributor 1 | Separator 4 | LOW | 201443 | 08.07.2012 | 08:51:36 | OPEN | 120708 54 | WA |
| WARNING | I | Cleaning | Watertank 3 | LOW | 740874 | 08.07.2012 | 08:45:12 | OK | 120708 53 | WA |
| WARNING | I | Cleaning | Watertank 7 | LOW | 740882 | 08.07.2012 | 08:40:46 | OPEN | 120708 52 | WA |
| INFO | - | Engine | Ignition | MAINT | 307663 | 08.07.2012 | 08:25:50 | | 120708 51 | IN |
| WARNING | 0 | Distributor | Motor 3 | NORM | 200486 | 08.07.2012 | 08:17:03 | OK | 120708 50 | WA |
| INFO | - | Drive 3 | Lubrication | MAINT | 540600 | 08.07.2012 | 08:02:13 | | 120708 49 | IN |
| WARNING | I | Packaging | Foil | LOW | 730229 | 08.07.2012 | 08:01:30 | OPEN | 120708 48 | WA |
| WARNING | I | Distributor | Motor 3 | SLOW | 200486 | 08.07.2012 | 07:59:04 | OK | 120708 47 | WA |
| WARNING | I | Packaging | Pressure | LOW | 730800 | 08.07.2012 | 07:48:45 | OK | 120708 46 | WA |
| ALARM | 0 | Exhaust | Filter | CLEAR | 309550 | 08.07.2012 | 07:39:02 | OK | 120708 45 | AL |
| WARNING | I | Drive 3 | Lubrication | LOW | 540600 | 08.07.2012 | 07:30:01 | OK | 120708 44 | WA |
| ALARM | I | Distributor 2 | Break | HOT | 200309 | 08.07.2012 | 07:15:10 | OK | 120708 43 | AL |

Fig. 9. A mock-up of an alarm list for a plant control system (integrated from a spreadsheet into UsER by graphics import within a rich text section)

### *7.2 Storyboards*

Storyboards are patterns of usage, i.e. scenarios along with sketches of the human-machine system in place. Operators will be described who are using the system within their work environment. Storyboards may be cartoon-like sketches depicting operators using their displays and controls during a specific task. It might for example show a plant operator setting up a production line step by step while communicating with technicians inside the plant.

In certain cases storyboards might help to understand critical incidents or accidents by describing relevant phases and activities of a team of operators before, during or after some critical event. This will illustrate critical paths during the operation of a control system and give hints about improvement in the procedures, work spaces or especially controls, displays or communication devices.

Currently storyboards may be created through the scenario module. In later versions of UsER a special module with support for graphic storylines will be added.

### *7.3 Styleguides*

Any industrial development will be done within certain design rules. Especially human-machine interfaces are defined and implemented within a design framework. These rules and design patterns will be defined in form of a *user interface styleguide*.

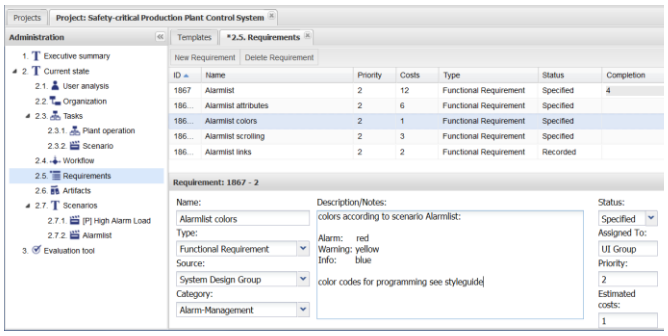
There are many types and forms of user interface styleguides. UsER will provide a toolkit for user interface styleguides that allows to describe any necessary details of design on the senso-motorical level (e.g. buttons, knobs, levers), the lexical level (e.g. icons, notions, acronyms), the syntactical level (e.g. drag&drop, input methods, display structures), the semantical level (e.g. mapping of colors and symbols to system states) and the procedural level (e.g. typical operating sequences like activating or deactivating an automatic function).

Styleguides may already be supported at least partially in the current mock-up environment to only allow interaction design along the defined styleguide. A complete user interface styleguide module for UsER is under development.

## 8. CLAIMS, REQUIREMENTS AND FEATURES

Analysis and design have to clarify a system from its early ideas and goals (claims), through well defined requests (requirements) into implementable functions (features). Developers have to be able to track and trace from claims to requirements to features and vice versa for clarification and rationales.

UsER enables the analyst, designer or evaluator to formally create, track and refine such claims, requirements and features as part of the clarification and specification process (Fig. 10).



| ID | Name | Priority | Costs | Type | Status | Completion |
|---|---|---|---|---|---|---|
| 1867 | Alarmlist | 2 | 12 | Functional Requirement | Specified | 4 |
| 186... | Alarmlist attributes | 2 | 6 | Functional Requirement | Specified | |
| 186... | Alarmlist colors | 2 | 1 | Functional Requirement | Specified | |
| 186... | Alarmlist scrolling | 2 | 3 | Functional Requirement | Specified | |
| 186... | Alarmlist links | 2 | 2 | Functional Requirement | Recorded | |

Requirement: 1867 - 2

Name: Alarmlist colors
Type: Functional Requirement
Source: System Design Group
Category: Alarm-Management

Description/Notes:
colors according to scenario Alarmlist:

Alarm: red
Warning: yellow
Info: blue

color codes for programming see styleguide

Status: Specified
Assigned To: UI Group
Priority: 2
Estimated costs: 1

Fig. 10. Hierarchical requirements structures in UsER with IDs, descriptions, states, priorities and project-specific grouped categories

### *8.1 Claims*

*Claims* define the basic ideas and goals of a system. They are often discussed and seldom documented in real development processes. UsER allows defining them early and rough, as well as more elaborated claims and mapping them into requirements as soon as possible. Claims may be visions, ideas, requests, assumptions or just beliefs about the system and operational organization to be developed.

### *8.2 Requirements*

The definition of *requirements* is the very center of systems engineering. Requirements are the unique reference points for contracts as well as releases of systems. Requirements have to be fulfilled by the solutions. It has to be testable and decidable whether they are reached or not.

Requirements defined within UsER may be exported through standardized ReqIF-format, spreadsheet tables or CSV-format into other development environments and vice versa.

### *8.3 Features*

For a solution, requirements have to be transformed into *features*. Designs and implementations are solutions for requirements. Features may be defined by short descriptions, by mock-ups or formal descriptions depending on the complexity and development method.

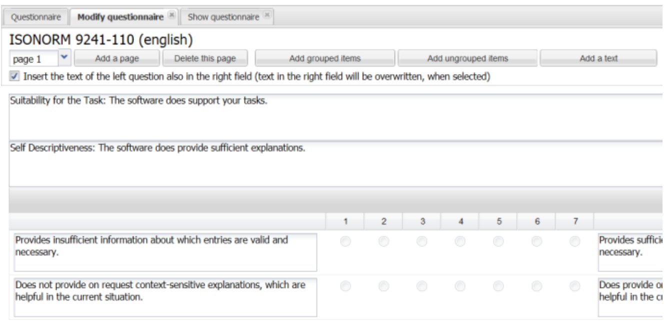
UsER will first of all give features unique names and enable the developers to cross-reference them with requirements, mock-ups, scenarios or external descriptions inside other documents or tools.

## 9. EVALUATION

During and after the development process, solutions have to be checked for their appropriateness and quality, especially for their usability. For this purpose, UsER provides an evaluation module consisting of a collection of usability evaluation methods and tools, like ISONORM (dialogue criteria conformance test for ISO 9241-110), SUS (System Usability Scale; a simple usability questionnaire), or NASA

TLX and SEA (both for workload evaluation). Other methods can easily be added into the framework as needed.

Questionnaires may be refined from a template archive and activated to be used via a web-based interface for the interviewees with the given data inputs being collected, processed and displayed within UsER (Fig. 11).



ISONORM 9241-110 (english)

☑ Insert the text of the left question also in the right field (text in the right field will be overwritten, when selected)

Suitability for the Task: The software does support your tasks.

Self Descriptiveness: The software does provide sufficient explanations.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Provides insufficient information about which entries are valid and necessary. | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Provides suffici necessary. |
| Does not provide on request context-sensitive explanations, which are helpful in the current situation. | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Does provide o helpful in the c |

Fig. 11. Setting up a usability survey with 7-point Likert-scales for an ISO 9241-110 evaluation (detail)

### 9.1 Formative Evaluation

Formative evaluation can be performed within UsER at any time during the development process. This allows for example to test the usability of a screen design even in a quite early phase of development based on some mock-ups. Results will be associated with the design itself to decide about changes or the release of the design for the next step.

### 9.2 Summative Evaluation

Summative usability evaluation is the final step before the release of a system. Application experts, customers and especially operators will be using the system providing their evaluations and feedback to be able to decide about the release status. The results will stay inside UsER, connected to the special parts tested and available for improvement for later releases or different customers. Summative evaluation is a user-centered type of quality assurance concerning the usability of the system prepared for deployment.

## 10. DEVELOPMENT APPROACHES

Process control systems have to support the performance of regular tasks as well as the timely reaction to expected or even unexpected events. UsER supports the development in respect to both operational situations.

### 10.1 Task-Oriented Development

In highly defined work situations there will be predefined tasks that have to be performed by the operators under well defined circumstances. The development of task-oriented systems means mainly to do a task and role analysis and set up standard operating procedures (SOPs) based on the work breakdown structure.

UsER may be used for task-based operations in the following way:

a) specify goals in the form of claims;
b) create a hierarchical task model (HTA);
c) define information objects and artifacts;
d) assign tasks, information and work objects to organizational roles, positions or automation;
e) set up standard operating procedures (SOPs) through the task model;
f) create user models as user classes, stereotypes or personas;
g) set up the requirements list;
h) design the human-machine interface as mock-ups;
i) develop refined user interfaces with user interface builder or graphics editors;
j) define system features and a roll-out plan (revisions);
k) implement the human-machine system;
l) evaluate the human-machine with evaluation instruments like evaluation of time-to-complete tests for the SOPs, dialogue principles or load index depending on the operational requirements and circumstances.

The development of task-based system is the standard approach in usability engineering and work place design. It requires a solid analysis and understanding of the underlying work system as well as the accompanying economical factors. Even if they mainly address the standard operations, safety issues have to be incorporated into the analysis and design methods reflected as well by the SOPs and human-machine interfaces derived.

### 10.2 Event-Oriented Development

In case of critical events during operations, another development approach is known to be helpful. Based on the analysis of failure modes (FMEA) or the study of incident and accidents reports, special critical operational circumstances and situations will be envisioned. The planned or the current system will be checked for its appropriateness and as a result of this analysis system revisions or Emergency Standard Operating Procedures (ESOPs) need to be derived.

With the help of UsER the following development process might be applied for event-based operations:

a) take an FMEA, an incident or accident analysis report and define problem scenarios;
b) check the information and work objects (displays and controls) available to serve the situation and describe an interaction scenario;
c) identify weaknesses in the existing system by annotating tasks, responsibilities (roles), screenshots or storyboards;
d) check and improve the task tree as well as the organization, its roles as well as automation for appropriate assignment of responsibility or team work;
e) develop an improved system by mock-ups and discussions of the mock-ups with personas, real operators or domain experts;

f) derive emergency standard operating procedures (ESOPs);
g) set up new requirements;
h) create detailed human-machine interfaces with a graphics editor or other mock-up tool;
i) transform the requirements into functional features or functional changes;
j) implement the optimized system;
k) evaluate the new system revision with comparative evaluations like A/B tests, ergonomics evaluation or load index and compare to previous revisions.

The development of event-oriented control systems is an ongoing process during operations. Especially incident reporting and simulator training reveals important input for iterative improvements. Event-oriented development has to be as independent as possible from economical requirements in the sense of *Resilience Engineering* (Hollnagel et al. 2006; Hollnagel, 2009). It addresses abnormal situations, as well as faulty technology or behavior (Reason, 1990; Dekker, 2006, 2007). It will be important for the safety of a system close to or even outside the standard operations boundaries.

## 11. CONCLUSIONS AND FUTURE WORK

UsER is a framework, platform and repository for an integrated and modular development of human-machine systems with the special scope of usability engineering.

It supports the analysis, design, and evaluation of human-machine systems through interrelated analysis and design entities creating a meshed specification of the system to be developed. The meshed structure can be linearized for standard documentation and contracting. Various import and export interfaces allow online or offline connections to other development environments like software or requirements engineering frameworks.

The repository extends standard software and systems engineering environments by providing a broad variety of standard or specific methods of usability engineering. It enables the process and motivates the teams for user-centered development and user interface design thinking.

UsER has been implemented as a Java-, GWT- and MySQL-based advanced prototype with an open architecture that has already been used within the industrial development of ERP systems (business applications) as well as supervisory control systems (safety critical systems). As a modular system, UsER allows selecting modules for analysis, design or evaluation as needed for a specific development process and may be extended by the integration of new modules for special application development domains and contexts.

## REFERENCES

Annett, J. & Duncan, K.D. (1967). Task Analysis and Training Design. *Occupational Psychology*, 41, 211-221.

Carroll, J.M. & Olson, J.R. (1988). Mental Models in Human-Computer Interaction. In Helander, M. (Ed.), *Handbook of Human Computer Interaction.* Amsterdam: Elsevier, 45-65.

Cooper, A. (1999). *The Inmates are Running the Asylum.* Indianapolis: SAMS.

Dekker, S. (2006*). The Field Guide to Understanding Human Error.* Aldershot: Ashgate Publishing Ltd.

Dekker, S. (2007). *Just Culture.* Aldershot: Ashgate Publishing Ltd.

Herczeg, M. (2001). A Task Analysis and Design Framework for Management Systems and Decision Support Systems. *ACIS International Journal of Computer & Information Science,* 2(3), September, 127-138.

Herczeg, M. & Stein, M. (2012). Human Aspects of Information Ergonomics. In Stein, M. & Sandl, P. (Eds.), *Information Ergonomics*, Berlin Heidelberg: Springer, 59-98.

Hollnagel, E. (2009). *The ETTO Principle: Efficiency-Thoroughness Trade-Off.* Aldershot: Ashgate Publishing Ltd.

Hollnagel, E., Woods, D.D. & Levenson, N. (Eds.). (2006). *Resilience Engineering - Concepts and Precepts.* Aldershot: Ashgate Publishing Ltd.

John, B.E. (1995). Why GOMS. *ACM Interactions*. October 1995, 80-89.

Johnson-Laird, P.N. (1983). *Mental models. Towards a cognitive science of language, inference, and consciousness*. Cambridge, MA: Harvard University Press.

Rasmussen, J. (1984). Strategies for State Identification and Diagnosis in Supervisory Control Tasks, and Design of Computer-Based Support Systems. In *Advances in Man-Machine Systems Research,* Vol. 1, 1984, 139-193.

Rasmussen, J. (1985). The Role of Hierarchical Knowledge Representation in Decisionmaking and System Management. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(2), 234-243.

Rasmussen, J., Pejtersen, A.M. & Goodstein, L.P. (1994). *Cognitive Systems Engineering*. New York: Wiley.

Reason, J. (1990). *Human Error.* New York: Cambridge University Press.

Rosson, M.B. & Carroll, J.M. (2002). *Usability Engineering. Scenario-Based Development of Human-Computer Interaction.* San Francisco: Morgan Kaufmann Publishers.

Shepherd, A. (1998). HTA as a framework for task analysis. *Ergonomics*, 41 (11), 1537-1552.

Stanton, N.A. (2006). Hierarchical task analysis: developments, applications and extensions. *Applied Ergonomics,* 37(1), 55-79.

Vicente, K.J. (1999). *Cognitive Work Analysis.* Hillsdale: Lawrence Erlbaum Associates.