Competencies for Informatics Systems and Modeling

Results of Qualitative Content Analysis of Expert Interviews

Johannes Magenheim, Wolfgang Nelles,
Thomas Rhode, Niclas Schaper
University of Paderborn
Paderborn, Germany
firstname.name@uni-paderborn.de

Sigrid Schubert,
Peer Stechert
University of Siegen
Siegen, Germany
firstname.name@uni-siegen.de

Abstract— **This article presents the first results of expert interviews conducted within the project MoKoM funded by the German Research Foundation (DFG). These are intended to refine our first theoretically derived competence model comprising competencies in the domains of informatics systems and modeling. In this context, the question arises, which software engineering workflows could be relevant for software engineering education at secondary schools and which are also adequate in professional contexts. Furthermore, it will be exemplarily shown which competence aspects have been added to the theoretical model and how to advance it in an empirical manner.**

General Issues in Engineering Education, Pedagogies, Modeling Competencies, System Comprehension, Active Learning

I. MOTIVATION

This article describes the methodical strategy to develop a competence model for informatics systems and modeling especially conducting and analyzing expert interviews within the project "Measurement Procedure for Informatics in Secondary Education (MoKoM)" funded by the German Research Foundation (DFG) [1]. Also, first exemplary results of the content analysis of the expert interviews are presented. Due to these results refinements on the theoretically derived model were accomplished. The article gives an outline of an exemplary analysis of 17 interview transcripts. The main focus is on the refinements of the theoretically derived competency model which could be obtained due to the results of the content analysis. Our temporary competence model was derived by means of theoretical and rational considerations. That means that curricula and expert papers treating informatics modeling and system comprehension were taken into account. However, a restriction on exclusively theoretically derived competencies would risk that the reference of competencies to complex requirements in real situations is neglected or disregarded. For this reason, an additional step is necessary in order to determine competencies more reliably, that is, ensuring an empirical access to determinate the relevant competencies. Conducting expert interviews within which the Critical Incident Technique was deployed, representing an appropriate empirical approach in order to detect the relevant competencies which belong to the two domains informatics modeling and system comprehension.

II. THEORETICALLY DERIVED COMPETENCE MODEL

In the following, the four competency dimensions of the informatics modeling and informatics comprehension model are presented.

K1. Basic Competencies

- K1.1 System Application
- K1.2 System Comprehension
- K1.3 System Development
  - K1.3.1 Business Modeling
  - K1.3.2 Requirements
  - K1.3.3 Analysis & Design
  - K1.3.4 Implementation
  - K1.3.5 Test
  - K1.3.6 Deployment

Figure 1. Competence Dimension K1 Basic Competencies

They form the fundamental framework of our theoretically derived competence model.

K2. Informatics views

- K2.1 External view
  - K2.1.1 Expectations of Systems' Behavior
  - K2.1.2 Informatics Literacy & Professional Practice
  - K2.1.3 Usability
- K2.2 Internal view
  - K2.2.1 Layered Architectures
  - K2.2.2 Net-Centric Computing
  - K2.2.3 Systems of Patterns
  - K2.2.4 Algorithms & Data Structures
  - K2.2.5 Fundamental Ideas of Computer Science
  - K2.2.6 Graphical Representations
  - K2.2.7 Languages (Programming & Modeling)
  - K2.2.8 Computational Thinking (imperative, functional, logical, object-oriented)

Figure 2. Competence Dimension K2 Informatics views

All competence dimensions were theoretically based on international syllabi and curricula with high reputation, e.g., the

978-1-4244-6571-2/10/$26.00 ©2010 IEEE

"Computing Curriculum 2001" of ACM and IEEE [2], "Model Curriculum for K-12 Computer Science" of the ACM [3]. These sources gave us a lot of inspiration and concrete advice for the structuring of our competence model. We also analyzed competence frameworks with high reputation, e.g., "The European Qualifications Framework (EQF)" of the European Union [4], "Definition and Selection of Competencies (DeSeCo)" of the OECD [5]. But none of these competence frameworks are really appropriate for education in the field of computer science.

The competence component K1.2 *System comprehension* was theoretically based on "Development of Competencies with Informatics Systems" [6]. The competence component K1.3 *System development* was theoretically derived from the process workflows of the Rational Unified Process (RUP) [7]. Each dimension comprises different competency components which characterize in more detail the requirements within the two mentioned informatics domains. According to Weinert´s notion of competency [8], competencies encompass both cognitive and non-cognitive skills and abilities.

K3. Complexity

- K3.1 Number of Components of Informatics Systems
- K3.2 Level of Net-Centric Computing
- K3.3 Stand-alone Systems vs. Distributed Systems
- K3.4 Level of Human-Computer Interaction

Figure 3. Competence Dimension K3 Complexity

Whereas the first two dimensions K1 *Basic Competencies* (see Figure 1) and K2 *Informatics views* (see Figure 2) are targeted on cognitive aspects with regard to the two domains, the fourth dimension K4 *Non-Cognitive Skills* (see Figure 4) includes attitudes, motivational, volitional and social skills which are of crucial importance for managing high demands and solving informatics problems in complex situations.

K4. Non-Cognitive Skills

- K4.1 Attitudes
  - K4.1.1 Perception & Anticipation of Informatics Systems
  - K4.1.2 Expectations for Informatics Literacy & Professional Practice
- K4.2 Social-Communicative Skills
  - K4.2.1 Cooperation & Teamwork
  - K4.2.2 Empathy: Change of Perspectives & Roles (User, Developer)
- K4.3 Motivational and Volitional Skills
  - K4.3.1 Openness to new Ideas & new Requirements
  - K4.3.2 Learning Motivation
  - K4.3.3 Commitment

Figure 4. Competence Dimension K4 Non-Cognitive Skills

The third dimension K3 *Complexity* (see Figure 3) is chosen in order to obtain a graduation of competencies which correspond to different levels of learners' skills. Also for this reason the first two dimensions are additionally graduated according to following taxonomic levels: (1) Knowledge, (2) Transfer and (3) Evaluation.

III. Examples of Qualitative Content Analysis of Expert Interviews

A. Procedure of Conducting and Analysing the Expert Interviews

In a first step of the empirical procedure the expert interviews were conducted. 30 experts on informatics, with a division in three equal groups were interviewed: experts in the domain of didactics of informatics, computer scientists and expert informatics teachers. This sample configuration was chosen in order to reveal the widest possible range of informatics and didactics expertise. The interview was based on a structured questionnaire manual and included questions concerning the expert status of the interviewees and several scenarios concerning application, testing, modifying and developing of informatics systems resp. software. Altogether 12 scenarios were used in the interview. To limit the duration of the interviews to an appropriate length each interviewee was only interviewed to four scenarios. Each interview lasted about 60 to 90 minutes. The interview included the following steps: (1) Welcoming the interviewee and introducing the underlying notion of competency and the interview technique (critical incident technique), (2) Presenting each scenario to the interviewee and encouraging him or her to give details of his/her intimate proceedings and approaches to solve the presented informatics problem, (3) Asking the interviewee to evaluate the four hypothetical scenarios with regard to their representativeness (with reference to rating scales).

The expert interviews were transcribed in full and analyzed by means of qualitative content analysis according to Mayring [9]. The main objective of qualitative content analysis is to reduce the large text material to a manageable size in such a manner that important information is not wasted. *Summarizing, explicating and structuring* are the main techniques of qualitative content analysis. The first technique enables to compress extensive proportions to text material. Deploying the second technique additional text material is brought to those passages in the text which need further explanation. The third technique means that a certain structure is extracted systematically from the text material. In our case this structure was brought to the text material in the form of our category system which belongs to our theoretical competency model. Analyzing the interview transcripts, all three techniques were applied. They do not mutually exclude one another, but rather complement each other well. At first, meaning units (i.e. textual elements containing relevant information concerning competencies) were systematically extracted. In the next step, it was investigated if the gathered meaning units could be matched to the competency categories of our model.

B. Informatics Systems

In Section B, we focus on K1.2 S*ystem comprehension* and its relations to other competence dimensions through concentration on a typical scenario "testing of unknown software (developed by others)". This scenario and its questions will be introduced to the interviewees as follows.

Scenario: "*You are asked by a colleague to test his software, which was developed to solve configuration problems, e.g., to set up a new car or a new computer.*"

Question 1: *"What is your strategy of testing to solve this problem? Which aspects do you have to bear in mind?"*

Question 2: *"Which cognitive skills are required for such a software exploration?"*

Question 2.1: *"Which informatics views are important for this task?"*

Question 2.2: *"Which complexity would you assign to this task?"*

Question 3: *"Are there any attitudes or social communicative and cooperative skills which are necessary to accomplish this?"*

Question 4: *"Which differences of competence levels would you expect between novices and experts?"*

Question 5: *"Could you imagine a potential pupil's procedure to solve this problem?"*

Question 6: *"Which obstacles would pupils have to cope with?"*

This scenario was interpreted through qualitative content analysis. Out of 26 interviews, eight experts were asked about such a scenario. The first results of the qualitative content analysis have to be structured according to the dimensions of the competence model. Relations between the competence components and meaning units in the interview have to be found and described (matching).

For this we use the following expressions: Answer plus number of competence component followed by the original statement of the expert (if necessary shortened). If the expert discovered a competence component, which is underestimated, this leads us to an additional answer. If the experts delivered a recommendation how to foster this needed competence component, then we add a more sophisticated description of the situation.

Firstly, K1.2 has to be proved. Therefore, a key question of the analysis process is how the experts support the assumption that testing of unknown software delivers an approach to informatics systems comprehension. The following three answers are examples of this proof of our assumptions:

Answer K1.2 (system comprehension): *"Then you need to understand fundamental mechanisms or relations or concepts, correctness of software, termination, knowledge about software ergonomics"*.

Answer K1.2 (system comprehension): *"I need a kind of model-view on the informatics system to explain the system's reactions that could be surprising. That means I need a model of processes behind the GUI"*.

Answer K1.2 (system comprehension): *"I will use my increasing system comprehension to plan the input-output relations."* The experts recommend for learners an unexpected exploration with non-systematical testing, which will fail for sure, followed by the successful systematic strategies of testing. But this should not be misunderstood as a guide line for school lessons.

The more experienced the experts were in the field of school informatics, the easier we could match their meaning units with the components of the theoretically founded competence model. This could be a hint that the competence model is not close enough related to the practice of software development. To remove doubts we trust on the evaluation of the next version of the competence model through the method of expert rating. Until then, we take this as an indicator that the rough structure of the theoretically founded competence model is appropriate.

To illustrate this, we give one example to each competence component beginning with the competence dimension K2 *Informatics views*:

Answer K2.1.1 (expectations of systems' behavior): *"You are not able to cope with a certain input mask if you don't know which input is expected."*

Answer K2.1.3 (usability): *"All parts of a Graphical User Interface have to be tested."*

Answer K2.1.4 (functional view): *"When the functionality and reliability of the informatics system was systematically tested, then we evaluate the other aspects of the quality of the software." (*new competence component; see Section IV/A*)*

Answer K2.2.8 (computational thinking): *"You have to learn which programming style is most appropriate for the given class of tasks. An important obstacle for this selection is the boundary of one and only one well-known programming style."*

The experts mention the handicap of the first and only successful problem solving strategy, e.g. a specific programming style. This phenomenon leads to failure, if one is matching a problem to an inappropriate problem solving strategy. A solution is a broad range of informatics strategies. The first successful informatics paradigm learned by a developer (student) is mostly applied even if it is not the best fitting solution.

Answer K4.1 (perception & anticipation of informatics systems): *"For the task of systematic testing perseverance and accuracy are indispensable for success."*

In turn, through the success you can improve your ability in accuracy and perseverance.

Answer K4.1.2 (expectations for informatics literacy & professional practice): *"A software engineer is responsible for the safety of humans and has to avoid any danger to life and limb."*

Answer K4.1.2 (expectations for informatics literacy & professional practice): *"Sensitization of pupils with respect to quality of software and responsibility."*

Answer K4.2 (social-communicative skills): *"One has to communicate with customers."*

Answer K4.2 (social-communicative skills): *"There is a serious contradiction between the competence of problem solving and the social-communicative competencies."*

It is necessary to supervise the development of social-communicative competencies, since they are not fostered as a side-effect of informatics problem solving. The task of systematic testing gives the opportunity to gain non-cognitive competencies on a higher level when the learner presents his results to other learners e.g. the explanation of use cases, the presentation of test results including the visualization of large data collections.

Answer K4.2.2 (empathy: change of perspectives & roles): *"When we test software of others, we have to learn to criticize in a fair and sensitive way."*

To foster empathy, systematic testing delivers the opportunity of mutual peer reviewing, i.e. each person has to give constructive criticism, handle criticism of others and not to insult others in a careless way while criticizing.

Answer K4.3 (motivational and volitional skills): *"This process leads to deep satisfaction with your own work and after a long period to a positive attitude towards high quality of software."*

From K1.2 to K4.3, the content analysis has shown 14 especially illuminating results in the form of support with refinements of the theoretically derived competence model. This leads us to the conclusion that the decision for this research method was very successful. The set of data allows a new version of the competence model influenced by the empirical studies (see Section IV / A).

C. Informatics Modeling

A typical scenario for K1.3 *System Development* in the expert interviews is the development of a software project or to pass a particular workflow within the software development process. In the following one exemplary scenario "Merchandise Management System" will be introduced.

Scenario "Merchandise Management System":

"You are asked to develop a software based merchandise management system for a small school kiosk.

Question 1: *"What is your course of action to solve this task? Which software engineering workflows do you have to pass?"*

Question 2*: "Which graphical models would you apply?"*

Question 2.1: *"Which informatics views are important for this task?"*

Question 2.2: *"Which complexity would you assign to this task?"*

Question 3*: "Which cognitive skills are required to develop such a software system?"*

Question 4: *"Could you imagine a potential pupil's procedure to solve this problem?"*

Question 5: *"Which attitudes, social communicative skills and motivational aspects are necessary to solve this problem?"*

In this connection, meaning units have to be derived from the interviewees' answers to *Question 1-5*. Furthermore potential relations between meaning units and components of the theoretically derived competency model have to be investigated.

Many interviewees stressed that K1.3 *System development* is an important ICT related competency and learners have to pass the workflows of professional software engineering processes.

Answer K1.3 (system development): *"We have to run through the workflows of the waterfall model."*

Answer K1.3 (system development) *"We have to pass the workflows of professional software development processes."*

The following interviewees' utterances accentuate specific software development workflows (K1.3.1 – 1.3.5) and competencies which could be allocated to them.

The K1.3.2 *Requirements* was mentioned five times. In this interrelation it was declared that learners have to be able to analyze use cases and to specify functional requirements.

Answer 1.3.2 (requirements): *"Analyze use cases to gain information about the way the system is used."*

Answer 1.3.2 (requirements): *"Specify functional requirements."*

The K1.3.3 *Analysis & Design* was mentioned by a large number of interviewees. It was stated that K1.3.3 *Analysis & Design* are processes which obey many iterations. In order to gain additional information to refine the theoretically based competency model it was expedient to subdivide these utterances to *Analysis* and *Design*. Nevertheless these workflows are strongly interwoven and will be mostly passed within several iterations [7, p. 2].

Answer K1.3.3 (analysis & design): *"Analysis Workflow has to be passed."*

Answer K1.3.3 (analysis & design): *"Object-oriented-Analysis has to be run through."*

Answer K1.3.3 (analysis & design): *"Within Analysis Workflow, learners have to envision the problem/task".*

Concerning the *Design* the interviewees mentioned that learners have to choose appropriate concepts and to develop their own conception to design an informatics system.

Answer K1.3.3 (analysis & design): *"Learners have to get familiar with several design-techniques."*

Answer K1.3.3 (analysis & design): *"Learners have to be enabled to choose an appropriate concept to design an informatics system."*

Answer K1.3.3 (analysis & design): *"They have to be able to derive potential program modules and tasks."*

Answer K1.3.3 (analysis & design): *"After all learners must get to know how to develop an own conception for designing the software."*

Irrespective of a potential distinction between *Analysis* and *Design* the interviewees listed several UML modeling techniques. This shows that learners have to be able to choose appropriate modeling techniques with respect to the actual software iteration.

Answer K1.3.3 (analysis & design): *"Learners have to apply UML-diagrams"*

Answer K1.3.3 (analysis & design): *"Develop class diagrams."*

Answer K1.3.3 (analysis & design): *"Identify potential classes."*

Answer K1.3.3 (analysis & design): *"Identify and allocate constituent parts of a class diagram, i.e. attributes, methods, associations and inheritance."*

Answer K1.3.3 (analysis & design): *"Applying state charts."*

Answer K1.3.3 (analysis & design): *"Applying deployment diagrams."*

The K1.3.4 *Implementation* was mentioned four times. In this context it was stressed that learners should be enabled to understand algorithms and to become familiar to an (Object-oriented-) programming language

Answer K1.3.4 (implementation): *"Understanding algorithms (recursive algorithms in particular)."*

Answer K1.3.4 (implementation)*: "We have to learn how to implement programs in a specific programming language and to cope with syntax errors."*

Answer K1.3.4 (implementation): *"Becoming familiar to Object-oriented-Programming."*

Answer K1.3.4 (implementation): *"Understanding the distinction between class-/object- attributes and –methods. "*

As mentioned in chapter A it is important to foster K4.2 *Social Communicative Skills* within informatics problem solving processes. Passing the *Implementation* offers the opportunity to develop software in teams.

Answer K4.2 (social-communicative skills): *"Programming in teams"*

In addition the interviewees mentioned that learners have to be able to integrate software modules (worked out by teams) into a comprehensive software system. Although the following meaning units could be allocated to the K1.3.4 *Implementation,* this might give advice to supplement the theoretically derived competence model by adding the component K1.3.6 *Deployment*.

Answer K1.3.4 (implementation): *"Synchronizing modules developed in teams."*

Answer K1.3.4 (implementation): *"Assembling modules developed in teams."*

Additionally it was stated that learners are faced with informatics system of a different grade of completion. Consequently the following meaning units might give us advice how to supplement the competency dimension K3 *Complexity of an Informatics System*.

Answer K1.3.4 (implementation): *"Learners must be able to develop software from the scratch."*

Answer K1.3.4 (implementation): *"Learners have to be enabled to get involved in an already existing informatics system and to re-engineer it."*

Answer K1.3.4 (implementation): *"They have to learn to analyze interfaces of an existing informatics system."*

The relevance of the K1.3.5 *Test* was declared by three interviewees.

Answer K1.3.5 (test): *"Testing and justifying the product's quality by comparing it to the (functional) requirements."*

Moreover the interviewees mentioned some well-established software testing procedures:

Answer K1.3.5 (test): *"Performing Blackbox-Testing."*

Answer K1.3.5 (test): *"Performing Whitebox-Testing."*

Answer K1.3.5 (test): *"Performing Regression-Testing."*

In summary this chapter illustrates several meaning units, which were derived from the interviewees' answers. These were structured with respect to the subcomponents of K1.2 *System Development*. According to this assignment of meaning units to the competence components (K1.2.1-K1.2.5), chapter IV-B will show how to refine the theoretically derived competency model, i.e. illustrating components which have to be justified, pointing out components which seem to be appropriate, showing interrelations to other competency dimensions and lining out components to supplement the model.

IV. EMPIRICALLY REFINED COMPETENCE MODEL

A. Informatics System

In this paragraph we discuss the consequences of the content analysis (see Section III / B). The competence component K1.2 *System comprehension* has to be refined with two sub components, K1.2.1 *Requirements* and K1.2.2 *Test* (see Figure 5). The following examples will illustrate this:

Answer K1.2.1 (requirements): *"You have to compare the real system's behavior with the definition of its requirements."*

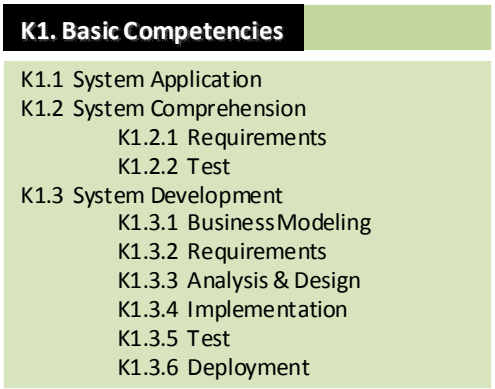

Figure 5. New Competence Dimension K1 Basic Competencies

Answer K1.2.2 (test): Systematic testing is mentioned by all experts. This should be learned through strategies (blackbox testing, whitebox testing), manuals and tools of testing.

The competence dimension K2 seems to be very important. A criterion for the success of problem solving is the selection of the most appropriate informatics strategy related to theoretical foundations and practical tools. Denning defines views as "The five windows of computing mechanics" [10, p 17]. Therefore, a change is recommended between different solution strategies of problem solving. Several experts stress the meaning of this relation. Informatics should not be reduced to programming. Therefore, we assume that this competence dimension is incomplete if only a static sequence of K2.1 *External view* and K2.2 *Internal view* with their facets is considered and we insert a new component, K2.3 *Change of views*.

Also the competence component K2.1 *External view* needs refinement, but only with one new sub component, K2.1.4 *Functional view* (see Figure 6), including input- output relations and state diagrams.

Answer K2.1.4 (functional view): *"In my opinion the view on the whole system has to be called functional view. This is connected with the analysis of input-output relations and designing and understanding of state diagrams. "*

For the competence dimension K3 we discovered a special situation. The competence components which we used, e.g., a "count of components", are not mentioned by any expert. But new competence components like combinatorial complexity and the complexity measures, time and space, are stressed very often. This leads us to the decision to insert two components, K3.5 *Combinatorial complexity* and K3.6 *Measures of complexity: time and space* (see Figure 7). Maybe we will later skip the unsuccessful components. It is too early to finalize this decision.

Answer K3.5 (combinatorial complexity): *"It is clear that we have to cope with the combinatorial complexity of 10^{10} solutions in the process of systematical testing."*

Answer K3.6 (measures of complexity: time and space): *"To characterize the quality of the software we should use runtime performance (including efficiency) and memory requirements to evaluate an informatics system, which lead us to the complexity measures, time and space."*

Furthermore, the scalability of problem solving strategies is related to the complexity of informatics systems. We found an impressive number of experts' statements, which connect scalability with the competence dimension K2 *Informatics views*. But in an unexpected way, the experts did not mention the role of scalability in the context of *Question 2.2* on the assigned complexity. We took all pro and contra arguments into consideration and came to the conclusion that it is necessary to put scalability into the competence model as new competence component, K3.7 *Scalability*, with strong connection to the other complexity factors.

To summarize the most important results of empirical evaluation line A (Informatics systems), in the following two paragraphs we focus on a deeper discussion of K2.3 *Change of view* (see Figure 6) and K3.7 *Scalability* (see Figure 7). Both results could be seen as a bridge between the empirical evaluation lines A and B (Informatics modeling).

K2.3 Change of view: In the particular research on system comprehension (during the last four years) we developed a successful strategy of changes of views from *observable behavior* (S_A) to *internal structure* (S_B) without focus on *construction of a concrete realization* (S_C). The main advantage is the change of view from *observable behavior* (S_A) to *combination of behavior and internal structure* (S_{AB}) and then to *combination of internal structure and construction of a concrete realization* (S_{BC}) and then to *combination of behavior and construction of a concrete realization* (S_{AC}). "The learners are able to explain, why a certain internal structure has been chosen with respect to the intended behavior of the informatics system." [11, p. 7]. Further research results are two important connections, firstly between static and dynamic properties of informatics systems and secondly between exploration and development of informatics systems. On our way from a theoretically established competence model to an empirically proved one, the change of view effects also the networked thinking in different ways: networking of fundamental ideas of informatics, networking of informatics sub systems, networking of successful patterns of informatics, such as object-oriented design patterns and architectural patterns. Surprisingly this opens an approach to informatics competencies not only connected with modeling and programming but to exploration of informatics systems.

K2. Informatics views

- K2.1 External view
  - K2.1.1 Expectations of Systems' Behavior
  - K2.1.2 Informatics Literacy & Professional Practice
  - K2.1.3 Usability
  - K2.1.4 Functional View
- K2.2 Internal view
  - K2.2.1 Layered Architectures
  - K2.2.2 Net-Centric Computing
  - K2.2.3 Systems of Patterns
  - K2.2.4 Algorithms & Data Structures
  - K2.2.5 Fundamental Ideas of Computer Science
  - K2.2.6 Graphical Representations
  - K2.2.7 Languages (Programming & Modeling)
  - K2.2.8 Computational Thinking (imperative, functional, logical, object-oriented)
- K2.3 Change of View

Figure 6. New Competence Dimension K2 Informatics views

K3.7 Scalability: Here we face the problem that most research results target on informatics modeling. We will refer to selected examples: In Informatics complexity of a problem resp. task plays an important role. Depending on the size of input n, how many resources, represented by the function f(n) are necessary to process a program on a machine. Resources may be the necessary time and space, the number accesses to the data bases etc. Some standard algorithms deliver good results for a small input size, but for a large input size they deliver bad results or no solution in a given time boundary. There-

fore, the criterion scalability can be used to decide, which solution (system, algorithm, process) should be applied to different problem sets.

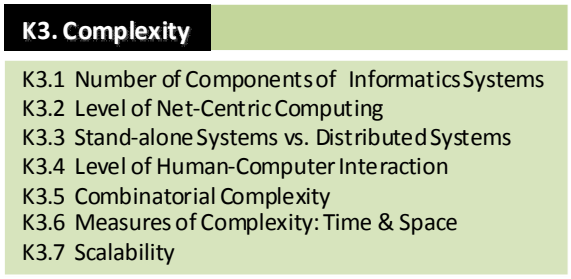

Figure 7. New Competence Dimension K2 Informatics views

There are nearly no publications for the influence of scalability of system comprehension. One example would be to find an efficient search strategy in relation to the expected search space. This means, you should not use the same strategy for searching key words in a finite text and for application of a web based search engine assuming a huge amount of candidates in the solution set. This leads us to heuristic solution strategies. Furthermore, scalability is an evaluation criterion for the range of application of a given informatics systems or tool.

The application of the empirical method Critical Incident Technique and content analysis established seven new competence components with respect to a scenario on system comprehension:

- K1.2.1 *Requirements,*
- K1.2.2 *Test,*
- K2.1.4 *Functional view,*
- K2.3 *Change of view,*
- K3.5 *Combinatorial complexity,*
- K3.6 *Measures of complexity: time and space,*
- K3.7 *Scalability.*

The same empirical method allows removing competence components. But to be one the safe side, we decided that we use the second empirical method, expert rating, to prove the agreement of the experts with the now more completed competence model. And after this research phase, it is easier to discover redundancies and removable components. This means, we can merge different components if necessary, in case of redundancies.

B. Informatics Modeling

The component K1.3 *System development* is an important part of the competency model, because it covers essential processes and competencies to develop an informatics system. Nevertheless the analyses of the expert interviews have shown the need to refine this component.

In this connection no interviewee stressed the K1.3.1 *Business modeling* or competencies which could be explicitly allocated to it. Hence this competency component has to be justified by taking further interviews and expert ratings into account or regarded as not be relevant for system development at least on secondary level.

K1.3.2 *Requirements* was mentioned several times and seems to be appropriate at this stage of work.

K1.3.3 *Analysis & design* was stressed by many interviewees. Nevertheless there are a lot of meaning units derived from the interviewee's utterances, which could be explicitly allocated to analysis or to design. Consequently it seems to be serviceable to split K1.3.3 *Analysis & design* into these separated workflows in order to categorize competencies more precisely. As mentioned before these workflows are interwoven and will be passed within several iterations [7, p. 2]. Irrespectively the analysis of the interviewee's utterances has shown that learners have to be fostered to choose serviceable modeling techniques and an appropriate course of action with respect to the actual iteration of the software engineering process (abbreviation: *sequencing pattern*). Fostering this kind of competency enables learners to restructure knowledge in multiple ways according to changing situational demands [12] and constitutes a precondition for situational learning. The application of different modeling techniques requires static views as well as dynamic views, e.g., class diagrams to support the static view and state diagrams for the dynamic view. To foster appropriate modeling competencies, K2.3 *Change of views* is inevitable.

Additionally, adding "Sequencing pattern*"* could be another purposeful refinement of the competency model. Furthermore, the selection of appropriate modeling techniques according to the actual stage of development of the software engineering process represents the appliance of scalable informatics problem solving strategies. Hence, K3.7 *Scalability* seems to be interwoven with other complexity factors of an informatics system and justifies the supplementation of K3 *Complexity* once more. Choosing expedient modeling techniques also comprises the appliance of several UML diagrams. This competence facet could be assigned to K1.3.3 *Analysis & Design* as well as to K2.2.6 *Graphical representations* and illustrates the linkage between K1 *Basic competencies* and K2 *Informatics views*.

As illustrated in chapter III / C K1.2.4 *Implementation* is an essential workflow within software engineering processes. The illustrated meaning units show its interrelation to other dimensions of the competency model. In this connection *understanding of algorithms* can also be assigned to K2.2.4 *Algorithms & data structures* and constitutes an important internal view on an informatics system. In addition to K1.2.4 *Implementation,* object-oriented programming can also be allocated to K2.2.7 *Languages (programming & modeling)*. These interrelations illustrate the linkage between K1 *Basic competencies* and K2 *Informatics views*. Furthermore the interviewees mentioned that learners have to be fostered to develop an informatics system from the scratch and to re-engineer an already existing informatics system, i.e. analyzing interfaces or integrating new program modules into a comprehensive system. These different approaches might illustrate the handling with informatics

systems of a different grade of completion. Hence, "grade of completion" might be an appropriate competency component to describe informatics systems of varying complexity. Consequently this competency component might be useful to supplement K3 *Complexity*.

In addition to the interviewee's utterances concerning the system's completion many interviewees declared expressly that learners have to be enabled to synchronize and to assemble program modules (developed by teams) in order to integrate them into a comprehensive informatics system. Accordingly it seems to be expedient to supplement K1.3.2 *Requirements* to the competency model. Additionally synchronizing and assembling program modules demands K4.2 *Social-communicative skills*. In this context workgroups have to agree to common interfaces for instance. This fact shows, that K1 *Basic competencies* is also related to K4 *Non-cognitive skills*.

K1.3.5 *Test* was also mentioned by many interviewees and seems to be appropriate as a constituent part of the competency model.

The application of the empirical method Critical Incident Technique and content analysis established two new competence components with respect to a scenario on system development:

- K2.3 *Change of view,*
- K3.7 *Scalability.*

This chapter focuses the empirical refinements of the theoretically based competency component K1.3 *System development*. In this context almost any of the subcomponents (K1.3.1-K1.3.5) seem to be appropriate. Furthermore there are several interrelations between the competency dimensions (K1 to K2, K3, K4). In addition, it has shown to be serviceable to add new components to the theoretical model. Particularly adding "Sequencing pattern" indicates that informatics modeling covers competencies to consciously select problem solving strategies according to the respective context the learners are faced. The supplementation of "Deployment" shows that modules worked out by working groups have to be consolidated in an expedient manner. Consequently it is inevitable to foster K4.2 *Social-communicative skills* explicitly and they should not be treated as negligibility.

V. CONCLUSIONS

As highlights of this article we summarize two research results, which will influence our further work in a specific way. These are *Scalability* and *Change of view*.

Scalability was found through the different research perspectives, informatics systems and informatics modeling. One important conclusion in Section IV / A was the criterion scalability can be used to decide, which solution should be applied to different problem sets. Another important conclusion in Section IV / B was that the context specific selection of modeling techniques represents the application of scalable informatics problem solving strategies. For the reader it is obvious that the person who is faced with an informatics system plays different roles, firstly as an explorer of informatics systems, secondly as a developer of such systems. It would be possible to describe both phenomena with different expressions to distinguish these aspects of scalability. But we decided to combine the two conclusions in one more abstract competence component K3.7 *Scalability*, because both aspects constitute the appropriate problem solving strategy.

In the same way we found change of view through the different research perspectives, informatics systems and informatics modeling. The research perspective on informatics systems delivers the next conclusion; the change of view affects the networked thinking in different ways. The analogue conclusion from the perspective of informatics modeling leads to K2.3 *Change of views*, which is inevitable to consciously use different modeling techniques.

This article presented first exemplary results of an empirical approach to determine competencies with reference to the domains *Informatics System Comprehension* and *Informatics Modeling.* The described results suggest that refinements and supplements to the theoretically derived competence model can be accomplished. On the basis of 17 from a total of 30 interview transcripts, it has been shown how the interviews were conducted and analyzed. Furthermore, it has been demonstrated how the results were applied to prove the theoretically derived competence model and to refine and supplement the model.

The described empirical procedure to complement the theoretical model is nevertheless restricted: One methodological restriction includes that the relevant competence requirements cannot be actuated comprehensively by the used scenarios. So it is important that the scenarios contain at least typical and representative tasks and problems to solve. This was evaluated by the represenativeness ratings of the experts. Furthermore, the actions described by the informatics experts might not necessarily mirror their actual behavior in those scenarios because they could describe idealized actions to solve the problems in the scenarios. On this issue, the different orientations of expertise of the interviewees serve as a corrective to some extent. The deployment of the qualitative content analysis took place adhering to comprehensible, methodical rules and principles. Nevertheless, qualitative analyses include inevitably interpretative processes which might restrict the objectivity, reliability and validity of the described analyses.

So it is necessary to prove the resulting competence model in further empirical research steps focussing on the content and criteria validity of the model. In a first step informatics experts with different backgrounds of expertise shall evaluate the relevance, difficulty, representativeness and differentiation of the determined competencies to prove the content validity of the model. The analysis of the criteria validity of the model is focussed on the following question: Do the determined competencies describe the requirements of successful acting in a sufficient manner? In order to answer this question, instruments to measure the different facets of the competence model and the criteria behavior have to be developed. Based on these measurements, then the correlations between the levels of competence on the one hand and the successful problem solving be-

havior as criteria on the other hand can be interpreted as criteria validity indicators of the competence model.

REFERENCES

[1] C. Kollee, J. Magenheim, W. Nelles, T. Rhode, N. Schaper, S. Schubert, and P. Stechert, "Computer science education and key competencies," in 9th IFIP World Conference on Computers in Education – WCCE 2009, Bento Goncalves (Brazil), 2009. www.die.informatik.uni-siegen.de/e-publikationen/Publikationen/2009/WCCE2009_pap147.pdf

[2] J. Cross, P. Denning, "Computing Curriculum 2001," The Joint Curriculum Task Force IEEE-CS/ ACM Report. 2001, www.computer.org/education/cc2001

[3] Tucker, Allen (ed.), "A Model Curriculum for K-12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee,". 2nd Edition. New York, Association for Computing Machinery (ACM), 2006, www.csta.acm.org

[4] European Union, "The European Qualifications Framework (EQF)," 2008, ec.europa.eu/education/lifelong-learning-policy/doc44_en.htm

[5] OECD, "Definition and Selection of Competencies (DeSeCo)," Paris, OECD, 2005, www.oecd.org/dataoecd/47/61/35070367.pdf

[6] P. Stechert, "Fachdidaktische Diskussion von Informatiksystemen und der Kompetenzentwicklung im Informatikunterricht" (Development of Competencies with Informatics Systems). Commentarii informaticae didacticae. Vol. 2. Universitätsverlag Potsdam, 2009.

[7] Rational Software Corporation IBM. Rational unified process. best practices for software development teams, white paper, 1998

[8] F.E. Weinert, "Concept of Competence: A Conceptual Clarification". In: Rychen, D., Salganik, L., (Eds.) Defining and selecting key competencies, Hogrefe & Huber, 2001.

[9] P. Mayring, Qualitative Inhaltsanalyse. Beltz, Weinheim, 2003.

[10] P. Denning, "Great principles of computing," in Commun. ACM 46 Nr. 11, 2003, pp. 15-20.

[11] P. Stechert and S. Schubert, "A strategy to structure the learning process towards understanding of informatics systems," in Working / Joint IFIP-Conference Informatics, Mathematics and ICT (IMICT2007), Boston, USA, 2007. www.die.informatik.uni-siegen.de/e-publikationen/Publikationen/2007/2007_Boston_stechert_schubert.pdf

[12] R.J. Spiro, "Cognitive flexibility, constructivism and hypertext: Random access," in T. Duffy, & D. Jonassen, Constructivism and the Technology of Instruction. Hillsale, NJ: Erlbaum.