International Journal of Research Studies in Computing
2012 October, Volume 1 Number 2, 27-47

International Journal of Research Studies in
Computing
Volume 1 Number 1 April 2012

ISSN: 2243-772X
Online ISSN: 2243-7797

OPEN ACCESS

Content and process concepts relevant to computer science education: A cross-cultural study

Zendler, Andreas ✉
University of Education Ludwigsburg, Germany (zendler@ph-ludwigsburg.de)
McClung, O. William
Nebraska Wesleyan University, Lincoln, Nebraska, USA (mcclung@nebrwesleyan.edu)
Klaudt, Dieter
University of Education Ludwigsburg, Germany (klaudt@ph-ludwigsburg.de)

Received: 27 June 2012 **Revised**: 4 July 2012 **Accepted**: 5 July 2012
Available Online: 6 July 2012 **DOI:** 10.5861/ijrsc.2012.144

Abstract

The development of a K–12 computer science curriculum based on constructivist principles needs to be informed by knowledge of content and process concepts that are central to the discipline of computer science. These central concepts play an important role in multiple domains of computer science, can be taught on every intellectual level, will be relevant on the long term, and are related to everyday language and/or thinking. Two empirically based catalogues of central concepts of information science (content and process concepts) have recently been developed and validated for the German context. Examples of central content concepts are *problem, model,* and *algorithm;* examples of central process concepts are *analyzing, categorizing,* and *classifying.* Taking a cross-cultural approach, this study compares the combinations of content and process concepts identified as important in Germany with those considered relevant in the US context. Results show that (1) the combinations identified in Germany can be generalized to the US context, (2) other combinations can be identified in the US context that are also important in the German context, and (3) the combinations of content and process concepts identified in the two contexts can be integrated to generate a broader perspective that is valid for both contexts.

Keywords: computer science education; cross-cultural research; central content concepts; central process concepts

Content and process concepts relevant to computer science education: A cross-cultural study

1. Introduction

The significance of computer science for the economy and for society in general is undisputed (Fuchs & Hofkirchner, 2003; Bauer, 2009; Tucker & Wegner, 2009; Johnson & Miller, 2009). The *Towards 2020 Science* report (Emmott, 2006) highlights the growing importance of computer science at the interface with other branches of science and emphasizes the critical role of computer science concepts such as algorithm, process, and program/data across scientific fields: "Several fundamental computer science concepts are already on their way to becoming household names in science, and many more will follow" (Shapiro, Harel, Bishop, & Muggleton, 2006, p. 24). The authors illustrate the role of computer science concepts by reference to the example of the pair of concepts *program* and *data*: "(…) we expect core computer science concepts on interchangeability of program and data (…) to prove essential for the full understanding of the role of DNA as programs and data" (Shapiro et al., 2006, p. 25). Ericson (2008) identifies five reasons why computer science is becoming increasingly important in schools: (1) computer science leads to multiple career paths, (2) computer science is important intellectually across a variety of disciplines, (3) computer science is important to industry, (4) computer science supports and links to other sciences, and (5) computer science teaches both scientific and societal problem solving.

A key challenge facing those involved in computer science education (ACM, 2003; 2008) is to decide on the subject matter to be taught. The spectrum currently ranges from training in the use of computer software via programming courses to the solution of theoretical problems. Teachers' uncertainty is evident in the fact that course content tends to reflect current trends and developments and to draw on short-lived product knowledge. Given the rapid pace of development in the field of information technology, however, this knowledge soon becomes obsolete. Instead, computer science education should equip students with knowledge and skills that will remain relevant in the longer term, which they can use in their everyday lives, and that are to some extent representative of the subject. The contents to be covered in computer science education have previously been discussed primarily in the context of fundamental ideas (Schwill, 1994). According to Schwill, a fundamental idea is a scheme of thinking, action, description, or explanation that satisfies four criteria: It must be relevant in multiple domains of a discipline (horizontal criterion). It must be teachable on every intellectual level (vertical criterion). It must remain relevant in the longer term (time criterion). And it must be related to everyday language and/or thinking (sense criterion). Several scientists have proposed catalogues of the basic concepts or fundamental ideas of computer science (Nievergelt, 1980; 1990; Schwill, 1994; Denning, 2003; Loidl, Mühlbacher, & Schauer, 2005; Wursthorn, 2005; Armoni & Ginat, 2008).

In terms of validity, however, the published catalogues have a number of shortcomings: (1) they are based on the subjective judgments of a single author or small group of authors, (2) they lack empirical verification, (3) they relate only to some sub-domains of computer science, (4) their validity has been established only for the national context in which they were developed. We have addressed points (1) to (3) in previous research. Specifically, the results of the three studies by Zendler and Spannagel (2008), Zendler, Spannagel, and Klaudt (2008), as well as Zendler, Spannagel, and Klaudt (2011) were based on the judgments of a larger sample of experts, were empirically derived, and related to whole discipline of computer science.

In the current discussion on curriculum development in computer science education, the combination of two scientifically informed approaches is considered crucial: first, the *structure of the discipline* approach introduced by Bruner (1960); second, the *process as content* approach, based on the work of Parker and Rubin (1966), which has more recently enjoyed a renaissance thanks to the three books edited by Costa and Liebmann (1997a; 1997b; 1997c). In the study by Zendler, Spannagel, and Klaudt (2011), which drew on these two approaches and on a constructivist theory of learning (Ben-Ari, 2001; Hadjerrouit, 2005a; 2005b; Machaniak, 2007; Moreno, González, Castilla, González, & Sigut, 2007), 24 German computer science professors rated the relevance of 15 content concepts (e.g., *algorithm, problem,* and *model*) with respect to 16 process concepts (e.g., *analyzing, categorizing,* and *classifying*) on a 6-point scale from ("*no importance*") to 5 ("*great importance*"). The main finding of the study was that there are specific groups of content concepts that should be taught in combination with specific groups of process concepts. In total, 15 blocks of content and process concepts were identified as being particularly relevant (e.g., the blocks of the content concepts *problem*, *model, algorithm, structure, information,* and *data* in combination with the process concepts *categorizing, classifying, generalizing,* and *finding cause-and-effect relationships*).

This study builds on the findings of Zendler, Spannagel, and Klaudt (2011), which were obtained in the German context. Taking a cross-cultural approach (see Berry, Poortinga, Segal, & Dasen, 2002; Lerner, Easterbrooks, Mistry, & Weiner, 2003; Saraswathi, 2003), it aims to replicate the findings presented by Zendler, Spannagel, and Klaudt (2011) in a different national context, namely the United States. In so doing, the present study seeks to address the fourth limitation of previously published catalogues of computer science concepts identified above, namely that their validity has been established only for the national context in which they were developed. To our knowledge—as informed by a review of the relevant literature (e.g., Computer Science Education, IEEE Transactions on Education, ACM Transactions on Computing Education) and of conferences in the field of computer science education in the years 2000 to 2011—no previous studies have examined the validity of the available catalogues of computer science concepts from a cross-cultural perspective.

Rather, the few published articles presenting international studies on computer science education have focused on comparing the curricula implemented in Europe and the United States (Scime, 2008), on the internationalization of computer science education (Douglas, Farley, Lo, Proskurowski, & Young, 2010), and on aspects of team building in software development (Burnell, Priest, & Durrett, 2002; Weinberg, White, Karacal, Engel, & Hu, 2005; Egea, Kim, Andrews, & Behrens, 2010).

Based on the cross-cultural research paradigm proposed by Berry, Poortinga, Segal, & Dasen (2002, pp. 3–5), this study addresses three research goals:

A. *Transport and test goal*: Can the combinations of content and process concepts identified in the German context be generalized to the US context?

B. *Discover variations goal*: Can other combinations be identified in the US context that are also important in the German context?

C. *Assemble and integrate goal*: Can the combinations of content and process concepts identified in the two contexts be integrated to generate a broader perspective that is valid for both contexts?

On the basis of these three goals, we formulated the following research hypothesis:

"German computer science professors differ from US computer science professors in their evaluations of the relations between central content concepts and central process concepts of computer science".

In Section 2, we present the methods applied, describing the study design and procedures and the data analysis strategy. In Section 3, we give a detailed account of our findings. In Section 4, we discuss those findings and, finally, draw implications for the internationalization of the field of computer science.

2. Methods

2.1 Study Design

Study design. The hypotheses were tested in a SPF-2•15×16 split-plot design (3-factor design with repeated measures of factors *B* and *C*, see Figure 1; (Winer, Brown, & Michels, 1991; Kirk, 1994).

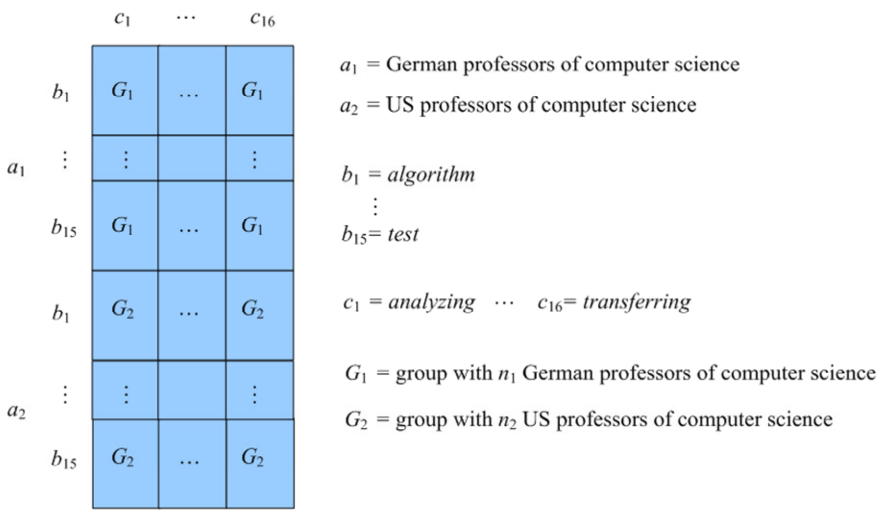
c_1
c_{16}
a_1
a_2
b_1
b_{15}
G_1
G_2
a_1 = German professors of computer science
a_2 = US professors of computer science
b_1 = *algorithm*
b_{15} = *test*
c_1 = *analyzing* ⋯ c_{16} = *transferring*
G_1 = group with n_1 German professors of computer science
G_2 = group with n_2 US professors of computer science

Figure 1. Layout of the 2•15×16 split-plot design

Independent variables. Factor *A* comprised the $p = 2$ groups surveyed, with factor level a_1 representing group G_1 of n_1 German professors of computer science and factor level a_2 representing group G_2 of n_2 US professors of computer science. Factor *B* represented the $q = 15$ content concepts $b_1, ..., b_{15}$: *algorithm, communication, computation, computer, data, information, language, model, problem, process, program, software, structure, system,* and *test*. Factor *C* represented the $r = 16$ process concepts $c_1, ..., c_{16}$: *analyzing, categorizing, classifying, collaborating, communicating, comparing, creating and inventing, finding cause-and-effect relationships, finding relationships, generalizing, investigating, ordering, presenting, problem solving and problem posing, questioning,* and *transferring*.

Dependent variable. The dependent variable was the respondents' evaluation of the importance of a specific process concept for a specific content concept. Ratings were given on a 6-point scale from 0 ("*no importance*") to 5 ("*great importance*").

Power analysis. A power calculation of type II—*N* as a function of power ($1-\beta$), Δ, and α—was used to determine the necessary sample size for the 2•15×16 split-plot design (Mueller & Barton, 1989; Mueller, LaVange, Ramey, & Ramey, 1992): With a power ($1-\beta$) of 0.99, only large effects ($\Delta = 0.80$) on the dependent variable being considered significant, and a significance level of $\alpha = 0.05$, a total sample of approximately $N^* = 30$ ($n_1^* = 15$ German professors of computer science, $n_2^* = 15$ US professors of computer science) would be required, based on the power computations of Mueller and Barton (1989) or Mueller, LaVange, Ramey, and Ramey (1992) for ε-corrected *F* tests.

Operational hypothesis. Given the study design and the above specification of the independent and dependent variables, the operational hypothesis of the study can be formulated as follows:

"German professors of computer science differ from US professors of computer science in their evaluations of the relations between central content concepts of computer science (*algorithm, communication, computation, computer, data, information, language, model, problem, process, program, software, structure, system, test*) and central process concepts of computer science (*analyzing, categorizing, classifying, collaborating, communicating, comparing, creating and inventing, finding cause-and-effect relationships, finding relationships, generalizing, investigating, ordering, presenting, problem solving and problem posing, questioning, transferring*), as operationalized by their rating on a 6-point scale of the importance of a specific process concept for a specific content concept".

2.2 Procedures

Samples. For the empirical study, a total of 120 computer science professors at German universities were contacted in 2008, and 120 computer science professors at universities in the US state of California (Berkeley, Davis, Irvine, Los Angeles, Riverside, San Diego, Santa Barbara, Santa Cruz) were contacted in 2011 and invited to complete a questionnaire (see Appendix) on computer science concepts.

Questionnaire. The questionnaire began with a short introduction, in which the 15 central content concepts and the 16 central process concepts were listed in tabular form in alphabetical order. In constructing the questionnaire, we followed recommendations for the development and translation of questionnaires to be used in cross-cultural studies (Harkness, 2003).

Evaluation task. The $q = 15$ content concepts and the $r = 16$ process concepts were then presented in alphabetical order in a matrix, with the content concepts in the rows and the process concepts in the columns. Participants were asked to indicate the relevance of each of the 15×16 = 240 combinations in the matrix: *Each cell represents a combination of a concept and a process and requires an integer from 0 (no importance) to 5 (great importance) indicating the relevance of the combination.* Participants filled in each cell on a 6-point scale from 0 ("*no importance*") to 5 ("*great importance*").

Return rate. To maximize the return rate, we mailed both samples the questionnaires in sealed, personalized envelopes, enclosing a pre-addressed return envelope franked with stamps showing flower designs (see Dillman, 2000) for recommendations on increasing return rates). The return rate for the German professors of computer science was 20.0% ($n_1 = 24$ valid questionnaires of 32 returned questionnaires), which can be considered reasonable for a postal survey (see Vaux & Briggs, 2005). The return rate for the US professors of computer science was 12.5% ($n_2 = 15$ valid questionnaires of 16 returned questionnaires).

2.3 Data Analysis

In analyzing our empirical data, we followed recommended procedures for cross-cultural research (van de Vijver & Leung, 1997; Harkness, van de Vijver, & Mohler, 2003): (1) First, we conducted a three-factor analysis of variance with repeated measures in accordance with the 2•15×16 split-plot design (see Winer, Brown, & Michels, 1991, chapter 7). We conducted an a posteriori comparison of means to test for effects of (2) the $A \times B$ interaction and (3) the $A \times B \times C$ interaction. (4) We then performed a cluster analysis with the aim of identifying groups of combinations of content and process concepts that provide a broader perspective (van der Vijver & Leung, 1997) which is valid for both the German and the US contexts. Data analyses were conducted using SPSS 17.0; the power analysis was computed with PASS 8.0.9.

3. Results

Figure 2 visualizes the mean ratings (see Appendix for the original data) obtained from the German professors of computer science (a_1; n_1=24) and the US professors of computer science (a_2; n_2=15) for each of the 15 × 16 combinations of content concepts × process concepts (repeated measures factors $B \times C$).

3.1 Results for the Transport and Test Goal

To examine whether the combinations of content and process concepts identified in the German context could be generalized to the US context, we formulated three statistical hypotheses, which were tested at the significance level of $\alpha = 0.05$.

Statistical hypotheses. The three null hypotheses were as follows:

A. The means of the content concepts μ_1 under a_1 (German professors of computer science) and μ_2 under a_2 (US professors of computer science) are equal, such that:

$$H_0\text{: } \mu_1 = \mu_2.$$

B. The means of the content concepts $\mu_{1\bullet1}$, $\mu_{1\bullet2}$, ..., $\mu_{2\bullet15}$ under the 2 • 15 levels of the factor combinations $A \times B$ are equal, such that:

$$H_0\text{: } \mu_{1\bullet1} = \mu_{1\bullet2} = \ldots = \mu_{2\bullet15}.$$

C. The means of the content concepts $\mu_{1\bullet1\times1}$, $\mu_{1\bullet1\times2}$, ..., $\mu_{2\bullet15\times16}$ under the 2 • 15 ×16 levels of the factor combinations $A \times B \times C$ are equal, such that:

$$H_0\text{: } \mu_{1\bullet1\times1} = \mu_{1\bullet1\times2} = \ldots = \mu_{2\bullet15\times16}.$$

Testing the statistical assumptions. For an analysis of variance of a split-plot design, the data must satisfy the condition of sphericity. This assumption was tested using Mauchly's W test for sphericity, with the test statistic W being compared to a chi-square distribution to assess the adequacy of the sphericity assumption. The assumption of sphericity was not met for either the content concepts (W=0.001, $\chi^2_{104} = 216.67$, $p < 0.001$) or the process concepts (W=0.002, $\chi^2_{119} = 206.37$, $p < 0.001$) at the α level of 0.05. In the further analyses, we therefore applied the ε correction of degrees of freedom proposed by Huynh and Feldt (1976).

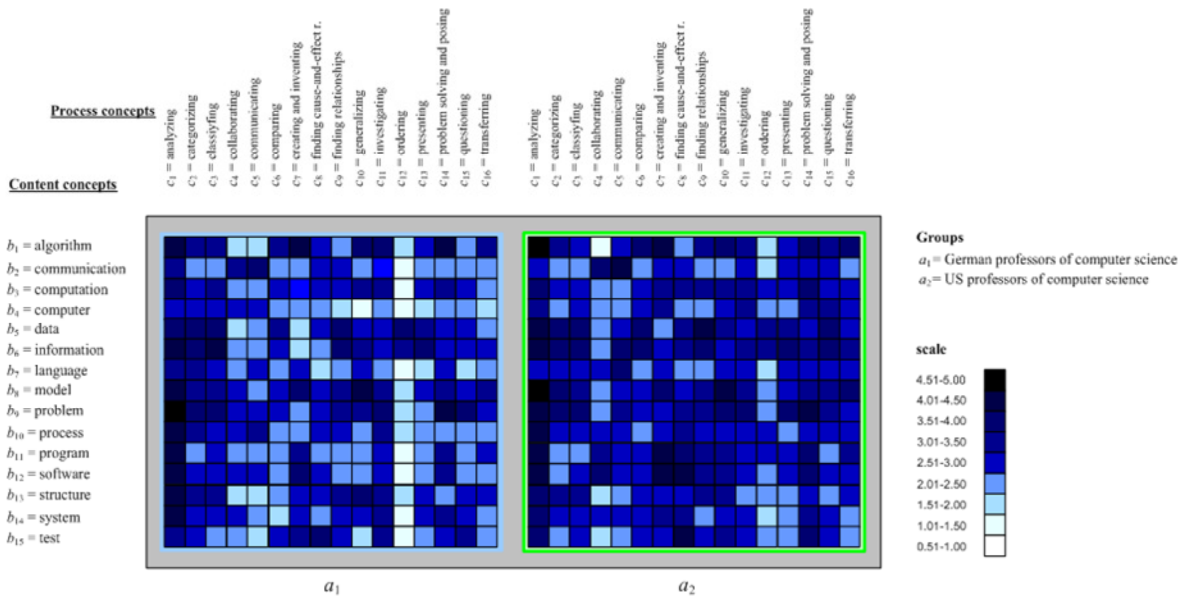

Figure 2. Mean ratings for all cells in the 2•15×16 split-plot design (n_1=24; n_2=15)

Table 1 presents the results of the ANOVA with Huynh–Feldt ε correction.

Table 1

Results of the ANOVA (with Huynh–Feldt ε correction of degrees of freedom)

Sources of variation	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
A	145.48	1	145.48	0.73	< 0.40
error (A)	7329.65	37	198.10		
Within subjects					
A x B	62.80	9	6.98	0.91	< 0.52
error (B)	2423.61	315	7.69		
A x B x C	314.83	65	4.84	1.27	< 0.07
error (B x C)	9133.85	2397	3.81		

The main effect *A* (German vs. US professors of computer science) was *not* significant at the α level of 0.05 ($F_{1,37}$= 0.73, $p < 0.40$). The corresponding H_0 was therefore *not* rejected: The German and US professors of computer science did not differ in their *global* evaluations of the content concepts. The interaction effect $A \times B$ (group × content concept) was *not* significant at the α level of 0.05 ($F_{9,315} = 0.91$, $p < 0.52$). The corresponding H_0 was therefore *not* rejected: The German and US professors of computer science did not differ in their evaluations of *individual* content concepts.

The interaction effect $A \times B \times C$ (group × content concept × process concept) was *not* significant at the α level of 0.05 ($F_{65,2397} = 1.27$, $p < 0.07$). The corresponding H_0 was therefore *not* rejected: The German and the US professors of computer science did not differ in their evaluations of the relationships between *individual* content concepts and *individual* process concepts. In summary, our findings for the *transport and test goal* indicate that the combinations of content and process concepts identified in the German context can indeed be generalized to the US context.

3.2 *Results for the Discover Variations Goal*

Individual Comparisons for the A × B Interactions

To examine whether it was possible to identify combinations of content and process concepts in the US context that are also relevant in the German context, we conducted mean comparisons of the group × content concept combinations. These comparisons were conducted using *t* tests to evaluate simple $\overline{AB}$ effects for *p*•*q*×*r* split-plot designs (Winer, Brown, & Michels, 1991, pp. 535–536), account taken of the ε correction of degrees of freedom ($df=df_{error\,(A)}+df_{error\,(B)}$=37+315=352; see Table 1) for the *t* tests. Figure 3 visualizes the means and standard errors of the $A \times B$ interactions. The results of these individual comparisons show that the German professors of computer science (a_1) and the US professors of computer science (a_2) did not differ significantly in their evaluations of content concepts at the α level of 0.05.

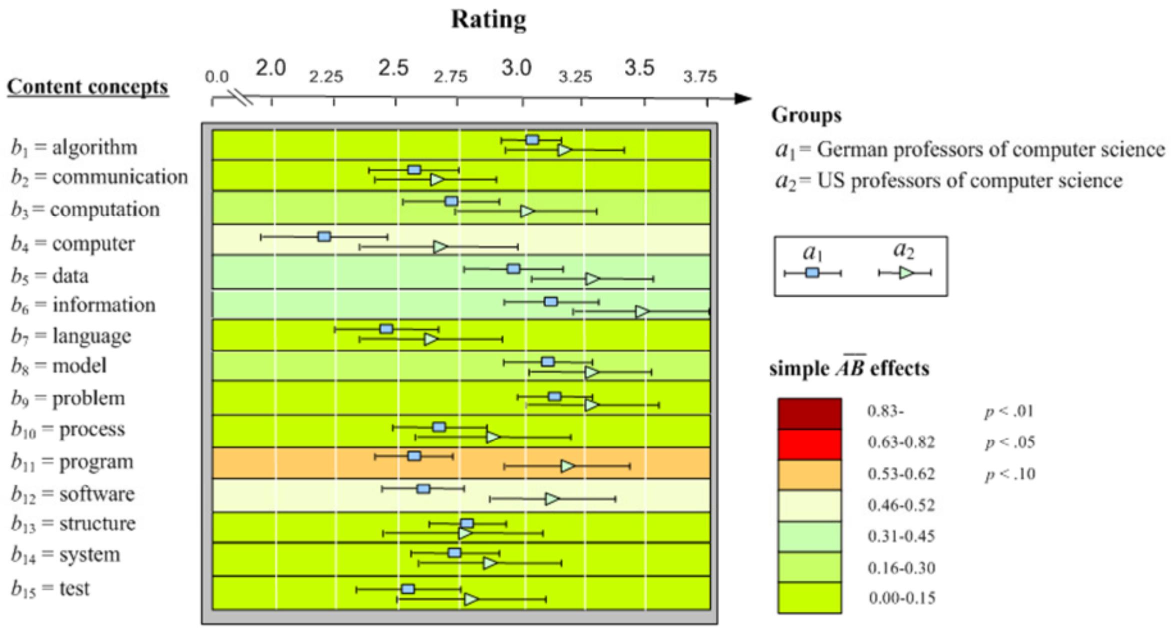

Figure 3. Individual comparisons for the factor level combinations $A \times B$

Individual Comparisons for the A × B × C Interaction

We next conducted a posteriori pair-wise mean comparisons to test which group × content concept × process concept combinations differed significantly, using $15 \times 16 = 240$ *t* tests to evaluate simple $\overline{ABC}$ effects for *p*•*q*×*r* split-plot designs (Winer, Brown, & Michels, 1991, pp. 535–536), account taken of the ε correction of degrees of freedom (df=$df_{error\,(A)}$+$df_{error\,(B)}$+$df_{error\,(C)}$+$df_{error\,(B\times C)}$=37+315+413+2397=3162 – see Table 1) for the t tests. Given the number of t tests that had to be conducted, an adjusted α level of 0.05/240 = 0.00021 was used to determine statistical significance.

Figure 4 identifies the significant $A \times B \times C$ interactions. As shown, the German professors of computer science (a_1) and the US professors of computer science (a_2) differed significantly in their evaluation of the relationships of *individual* content concepts with *individual* process concepts at the adjusted α level of 0.00021 ($t_{3240} \geq 3.72$, $p < 0.0002$).

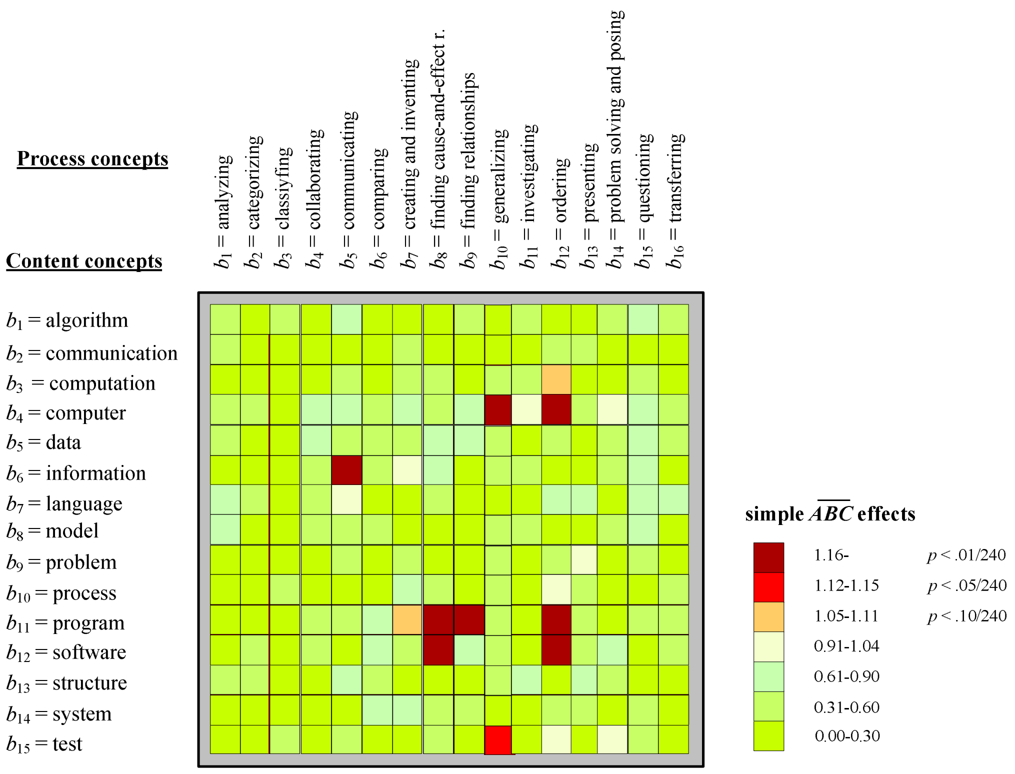

Figure 4. Individual comparisons for the factor level combinations $A \times B \times C$

In sum, our results for the *discover variations goal* showed that it was indeed possible to identify combinations of content and process concepts in the US context that are also important in the German context, as reflected by individual comparisons of the $A \times B \times C$ interaction. In particular, significant differences were detected in the two groups' evaluations of the relationships of the following content concepts with *individual* process concepts:

- *computer with generalizing and ordering;*
- *information with communicating;*
- *program with finding cause-and-effect relationships, finding relationships, and ordering;*
- *software with finding cause-and-effect relationships and ordering;*
- *test with generalizing.*

3.3 Findings for the Assemble and Integrate Goal

To address the *assemble and integrate goal*, we drew on a data set that incorporated the results of both the *transport and test goal* and the *discover variations goal*. This data set was generated by weighting the means according to their respective sample sizes (n_1=24 and n_2=15): the data obtained from the German professors of computer science were weighted by w_1=24/39; those obtained from the US professors of computer science, by

w_2=15/39. This approach takes account of the differences found with respect to the *discover variations goal* to the extent that the respective values are taken directly from the US data set (see Figure 5).

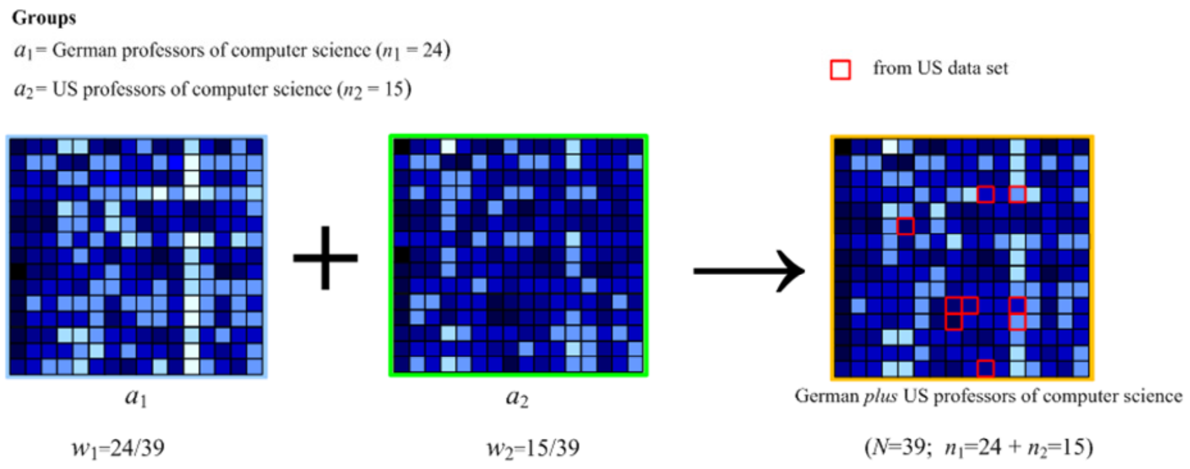
Groups
a_1 = German professors of computer science (n_1 = 24)
a_2 = US professors of computer science (n_2 = 15)
from US data set
a_1
a_2
German *plus* US professors of computer science
w_1=24/39
w_2=15/39
(N=39; n_1=24 + n_2=15)

Figure 5. Data set used to address the *assemble and integrate goal*

To examine whether the combinations of content and process concepts identified in the two contexts could be integrated to generate a broader perspective that is valid for both contexts, we first performed cluster analyses and then identified combinations of content and process clusters—so-called blocks—that are relevant to computer science education. In so doing, we drew on the data set with the weighted means (see Appendix for original data) of the German and US professors of computer science, as shown in Figure 6.

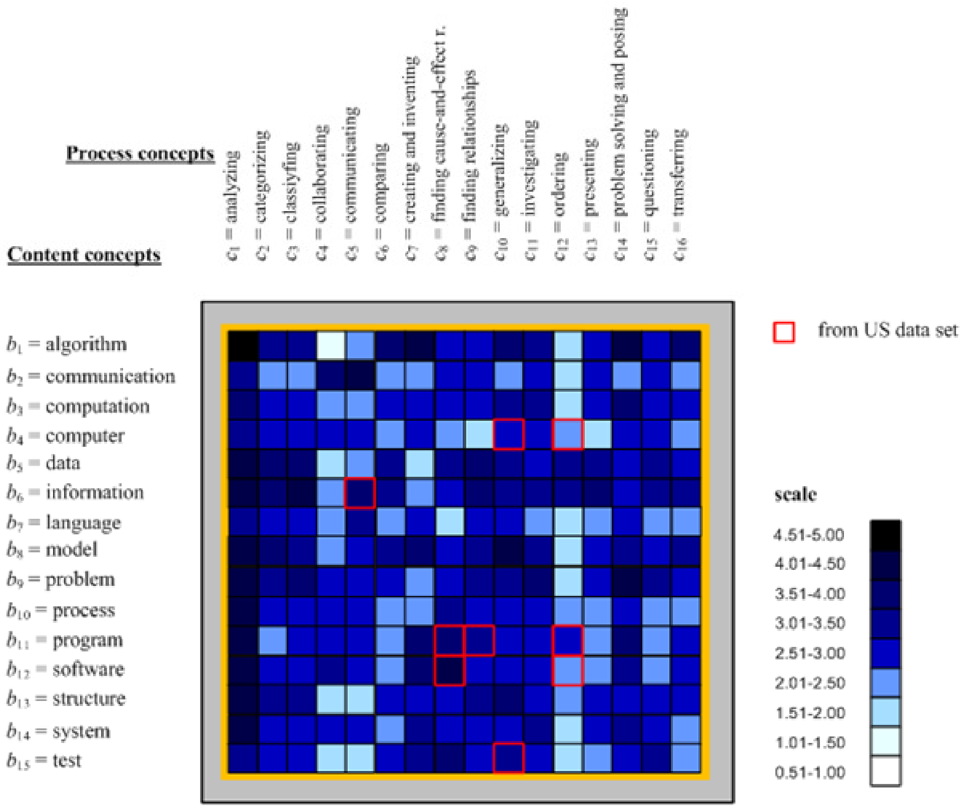

Figure 6. Mean ratings provided by the German and US professors of computer science (N=39; n1=24 + n2=15)

3.4 Cluster Analyses

The method of cluster analysis used was the procedure proposed by Ward (1963), with the squared Euclidean distance as the measure of distance (Everitt, Landau, & Leese, 2001). We used the C Index proposed by Hubert and Levin (1976) to determine the cut level. Two independent cluster analyses were performed: first, cluster analysis of the rows, with the content concepts as objects and the process concepts as criteria; second, cluster analysis of the columns, with the process concepts as objects and the content concepts as criteria. Figure 7 presents the two generalized cluster solutions for the content concepts and the process concepts. The heatmap (Grinstein, Trutschl, & Cvek, 2001) visualizes the mean expert ratings of the relevance of each content concept for each process concept (and vice versa). Furthermore, the two dendrograms in Figure 7 illustrate the clusters formed at each stage for both the content concepts and the process concepts.

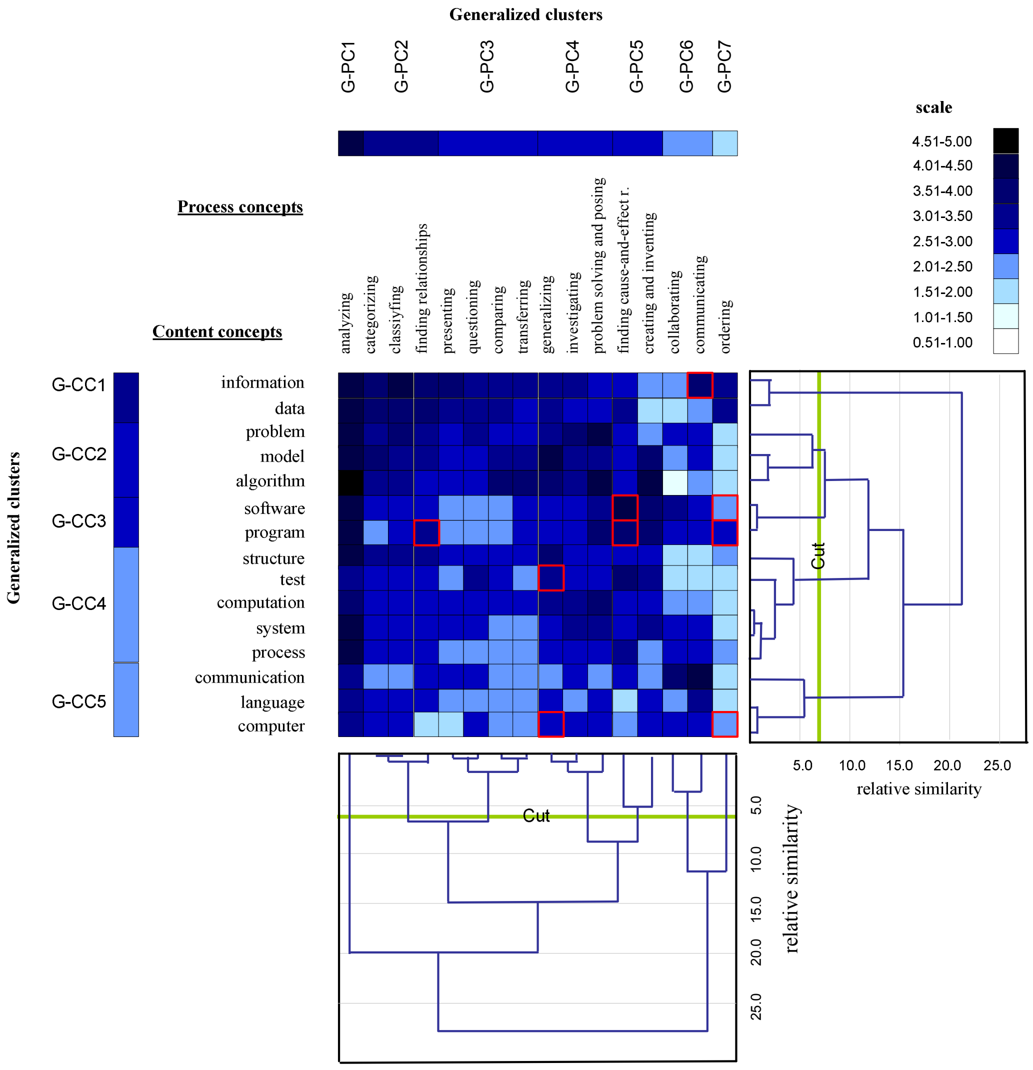

Figure 7. Cluster solutions for the content concepts and the process concepts ($N = 39$; $n_1=24 + n_2=15$)

Generalized clusters of content concepts

The cluster solution for the content concepts consists of the clusters *G-CC1, G-CC2, G-CC3, G-CC4,* and *G-CC5*.

A. *G-CC1 cluster*. This cluster contains the two content concepts *information* and *data*, which are characterized by very high values with respect to the process concepts *analyzing, categorizing,* *classifying,* and *finding relationships.* As the dendrogram shows, the two content concepts cannot be merged with any other content concepts, as their relative similarity values exceed the threshold value of the C index (see "Cut" in Figure 7).

B. *G-CC2 cluster.* This cluster comprises three content concepts—*problem, model,* and *algorithm*—which are characterized by very high values with respect to the process concepts *analyzing, categorizing, classifying,* and *finding relationships*, but relatively low values with respect to *creating and inventing* and *ordering.*

C. *G-CC3 cluster.* This cluster, which is the most homogeneous of the content clusters, consists of two content concepts: *software* and *program.* Both are characterized by a consistent profile across all process concepts. The very high values with respect to *finding-cause-and-effect relationships* and *creating and inventing* are especially notable.

D. *G-CC4 cluster.* This cluster is the one containing the most concepts, namely *structure, test, computation, system,* and *process.* It is characterized by the homogeneity of the expert ratings of the importance of these content concepts for the process concepts *categorizing, classifying,* and *finding relationships.* As the dendrogram shows, *computation, system,* and *process* form a cluster very early in the clustering procedure, with *test* and *structure* being added subsequently.

E. *G-CC5 cluster.* This cluster consists of three content concepts: *communication, language,* and *computer.* The content concepts in this cluster are characterized by low values with respect to almost all process concepts. The content concept *communication,* which forms a cluster with the other two concepts late in the clustering procedure, is distinguished by high values with respect to the process concepts *collaborating* and *communicating.*

Generalized clusters of process concepts

The cluster solution for the process concepts consists of the clusters *G-PC1, G-PC2, G-PC3, G-PC4, G-PC5, G-PC6,* and *G-PC7.*

A. *G-PC1 cluster.* This cluster consists of the process concept *analyzing*. It is characterized by very high values with respect to almost all content concepts (grand mean of the *G-PC1* cluster = 3.93). The peculiarity of this cluster—having only a single process concept—is reflected in the dendrogram: clustering with other clusters is not possible due to their high dissimilarity.

B. *G-PC2 cluster.* This cluster contains three process concepts: *categorizing, classifying,* and *finding relationships.* What is particularly notable in this homogeneous cluster is the very early grouping of the two process concepts *categorizing* and *classifying*, which have similar values with respect to almost all content concepts. The early clustering of *finding relationships* to the other two process concepts is also notable.

C. *G-PC3 cluster.* This cluster comprises four process concepts forming two sub-clusters: *presenting* and *questioning*, on the one hand, and *comparing* and *transferring*, on the other. Both sub-clusters could be identified early on the basis of their relative similarity coefficients (see Figure 6); each has a relatively consistent profile across the content concepts.

D. *G-PC4 cluster.* This cluster consists of three process concepts: *generalizing, investigating,* and *problem solving and posing.* They are characterized by high values with respect to almost all content concepts with the exception of *communication, language,* and *computer.*

E. *G-PC5 cluster.* This cluster comprises the two process concepts *finding-cause-and-effect relationships* and *creating and inventing.* The two concepts form a cluster relatively late; however, both are characterized by high values with respect to the content concepts *software* and *program.*

F. *G-PC6 cluster*. This cluster contains the two process concepts *collaborating* and *communicating*. These process concepts are characterized by similar profiles with respect to the content concepts *software, program*, *structure, test, computation, system,* and *process.*

G. *G-PC7 cluster.* This cluster consists of a single process concept, namely *ordering*. The cluster is characterized by low values with respect to almost all content concepts, on the one hand, but by high values with respect to the content concepts *information* and *data*, on the other.

3.5 Constructing Generalized Blocks of Content and Process Clusters

The aim of this part of the analysis is to identify sets of content and process concepts that should be taught in combination in computer science education. We used a pragmatic procedure that determines blocks as overlaps of clusters of content and process concepts. Specifically, blocks are identified as being relevant to computer science education if they are in the upper half of a median split. As illustrated in Figure 8, a total of 18 blocks were identified by means of median split [mean/median=2.88], namely G-CC1×G-PC1, G-CC1×G-PC2, G-CC1×G-PC3, G-CC1×G-PC4, G-CC1×G-PC7, G-CC2×G-PC1, G-CC2×G-PC2, G-CC2×G-PC3, G-CC2×G-PC4, G-CC2×G-PC5, G-CC3×G-PC1, G-CC3×G-PC4, G-CC3×G-PC5, G-CC4×G-PC1, G-CC4×G-PC2, G-CC4×G-PC4, G-CC5×G-PC1, and G-CC5×G-PC6.

A. *G-CC1×G-PC1 block.* This block contains the content concepts *information* and *data* in conjunction with the process concept *analyzing*. It thus emphasizes the importance of addressing information and data from the perspective of analysis in the computer science classroom.

B. *G-CC1×G-PC2 block.* This block comprises the content concepts *information* and *data* in combination with the process concepts *categorizing, classifying,* and *finding relationships.* Accordingly, it indicates that the content concepts information and data should be taught especially in the context of activities involving categorization.

C. *G-CC1×G-PC3 block.* This block contains the content concepts *information* and *data* together with the process concepts *presenting, questioning, comparing,* and *transferring.* As such, it suggests that the content concepts information and data should be approached from the perspective of presentation, questioning, comparison, and transfer to other situations.

D. *G-CC1×G-PC4 block.* This block combines the content concepts *information* and *data* with the process concepts *generalizing, investigating,* and *problem solving and posing.* Thus, it indicates that information and data can usefully be considered from the perspective of generalization, scientific investigation, and the definition, specification, description, and solution of problems in the computer science classroom.

E. *G-CC1×G-PC7 block.* This block contains the content concepts *information* and *data* in conjunction with the process concept *ordering*. It thus emphasizes activities in which information and data are systematically arranged on the basis of different criteria or relationships.

F. *G-CC2×G-PC1 block.* This block encompasses the content concepts *problem, model,* and *algorithm* together with the process concept *analyzing*. In terms of computer science instruction, it thus emphasizes the activities of analyzing problems, models, and algorithms.

G. *G-CC2×G-PC2 block.* This block comprises the content concepts *problem, model,* and *algorithm* in combination with the process concepts *categorizing, classifying,* and *finding relationships.* It thus indicates that problems, models, and algorithms should be approached from the perspective of categorization, classification, and generalization in the computer science classroom.

H. *G-CC2×G-PC3 block.* This block consists of the content concepts *problem, model,* and *algorithm* along with the process concepts *presenting, questioning, comparing,* and *transferring.* It thus underlines the importance of addressing problems, models, and algorithms in the context of activities involving presentation, questioning, comparison, and transfer to other situations.

I. *G-CC2×G-PC4 block.* This block contains the content concepts *problem, model,* and *algorithm* in combination with the process concepts *generalizing, investigating,* and *problem solving and posing.* In terms of computer science education, this block emphasizes the activities of investigating and generalizing problems, models, and algorithms.

J. *G-CC2×G-PC5 block.* This block consists of the content concepts *problem, model,* and *algorithm* along with the process concepts *finding cause-and-effect relationships* and *creating and inventing.* It thus emphasizes activities involving the identification of cause-and-effect relationships and the use of models in the creation and use of algorithms.

K. *G-CC3×G-PC1 block.* This block contains the content concepts *software* and *program* in combination with the process concept *analyzing.* It thus indicates that one focus of computer science education should be on the analysis and specification of needs and requirements in the early phases of software development.

L. *G-CC3×G-PC4 block.* This block contains the content concepts *software* and *program* together with the process concepts *generalizing, investigating,* and *problem solving and posing.* It thus underlines that the two content concepts should be taught in the context of activities involving generalization, investigating an area of interest, and defining and addressing problems.

M. *G-CC3×G-PC5 block.* This block consists of the content concepts *software* and *program* along with the process concepts *finding-cause-and-effect* and *creating and inventing.* It thus indicates that the two content concepts should be taught from the perspective of cause-and-effect relations and developing new ideas.

N. *G-CC4×G-PC1 block.* This block contains the content concepts *structure, test, computation, system,* and *process* in combination with the process concept *analyzing.* Accordingly, it emphasizes that the content concepts in this block should be taught in the context of activities focusing on analysis.

O. *G-CC4×G-PC2 block.* This block contains the content concepts *structure, test, computation, system,* and *process* together with the process concepts *categorizing, classifying,* and *finding relationships.* It thus highlights the importance of addressing problems, models, and algorithms from the perspective of categorization, classification, and generalization in computer science education.

P. *G-CC4×G-PC4 block.* This block comprises the content concepts *structure, test, computation, system,* and *process* in conjunction with the process concepts *generalizing, investigating,* and *problem solving and posing.* It thus focuses attention on generalization, scientific investigation, and the definition, specification, description, and solution of problems in computer science lessons.

Q. *G-CC5×G-PC1 block.* This block contains the content concepts *communication, language,* and *computer* along with the process concept *analyzing.* It thus emphasizes that these content concepts should be addressed in the context of activities with a focus on analysis.

R. *G-CC5×G-PC6 block.* This block contains the content concepts *communication, language,* and *computer* in combination with the process concepts *collaborating* and *communicating.* It thus emphasizes that coverage of these three content concepts in computer science classrooms should involve forms of collaboration (e.g., synchronous/asynchronous forms, local/distributed forms, horizontal/vertical forms) and communication (e.g., linguistic, visual, technological) in which the aim is to achieve shared goals.

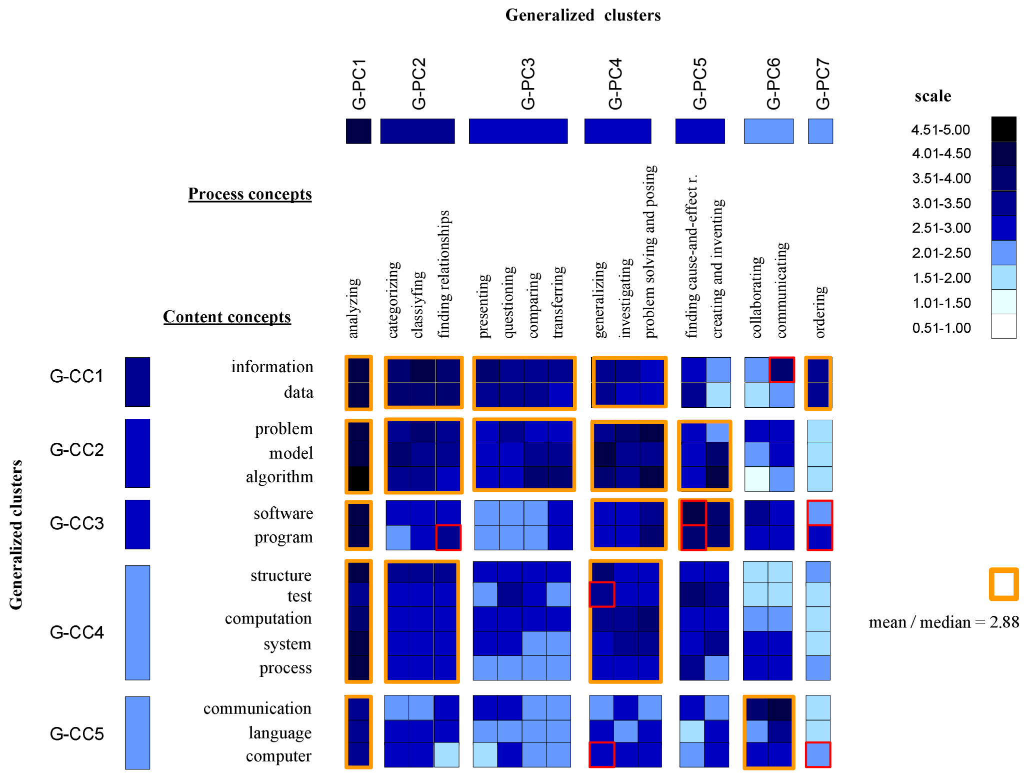

Figure 8. 5×7 matrix with 18 generalized blocks of content concepts and process concepts

4. Discussion

The first conclusion to be drawn is that our empirical findings do *not* support the hypothesis that German and US professors of computer science differ in their evaluations of the relations between central content concepts of computer science and central process concepts of computer science.

This study provides an empirical basis for identifying the combinations of content and process concepts that are relevant for K–12 computer science education in the international context. It thus contributes to the intercultural generalization of findings previously reported for Germany (Zendler & Spannagel, 2008; Zendler, Spannagel, & Klaudt, 2008; Zendler, Spannagel, & Klaudt, 2011).

The blocks of central computer science concepts identified in this study satisfy the demands of constructivist approaches to computer science education (Ben-Ari, 2001; Hadjerrouit, 2005a; 2005b; Machaniak, 2007; Moreno, González, Castilla, González, & Sigut, 2007)—in particular, because content concepts were combined with process concepts: “The most important issue for curriculum reform raised by the social construction theory is that we should look beyond content as defining curriculum, but consider also the way in which students encounter new knowledge as part of the curriculum” (Machaniak, 2007, p. 2).

The findings from the 5×7 matrix with its 18 generalized blocks of content and process concepts indicates that specific groups of content concepts should be addressed in combination with specific groups of process concepts in computer science classrooms. It is notable that the content concepts *data* and *information* have very similar values with respect to almost all process concepts. It thus seems that these two content concepts should be taught in combination with all of these process concepts. The same applies to the process concepts *communicating* and *collaborating*, which have very similar values with respect to almost all content concepts, and should thus be taught in combination with all of these content concepts. The 5×7 matrix further indicates that the content concepts *problem, model,* and *algorithm* have particularly high values with respect to the process concepts belonging to five clusters (*G-PC1, G-PC2, G-PC3, G-PC4, G-PC5*). Accordingly, these content concepts should be addressed in combination with a whole series of process concepts. In other words, these content concepts can be used to teach numerous process concepts in computer science education. The same applies to the process concept *analyzing* with respect to the teaching of content concepts. As the matrix shows, *analyzing* has high values with respect to all content concepts belonging to clusters *G-CC1* to *G-CC6*. In other words, *analyzing* is particularly important in computer science education when it is used to teach content concepts. The process concept *ordering*, which was rated to be important for content concepts *information* and *data* only, is also of particular interest.

Interpreting these results against the background of the cross-cultural research paradigm (Berry, Poortinga, Segal, & Dasen, 2002), its objectives and proposed methods (van der Vijver & Leung, 1997, chapters 2 and 4), we can draw the following conclusions: (1) The combinations of content and process concepts identified in the German context can indeed be generalized to the US context. (2) US professors of computer science evaluate the two content concepts *software* and *program* as being much more important than do their German counterparts. Moreover, it is possible to identify combinations of content and process concepts that the US professors of computer science evaluate differently from their German colleagues. (3) The combinations of content and process concepts identified in the US and German contexts can be integrated to generate a broader perspective that is valid for both contexts.

The results obtained have several implications for the internationalization of computer science education: In the development of K–12 computer science curricula to be used in the international context, it is particularly important to take aspects of communication into account, as indicated by blocks *G-CC5*×*G-PC1* and *G-CC5*×*G-PC6*. This finding is in line with the theorizing of Douglas, Farley, Lo, Proskurowski, and Young (2010, p. 412) on the actions needed to incorporate the impact of internationalization into the computer science curriculum. According to these authors, “The issues in cross-cultural communication are not unique to computer professionals, but are certainly important in the context of the global economy, multinational development teams, and localization of computer applications”.

5. Conclusions

The present findings are of great relevance for research-based approaches to the pre- and in-service education of computer science teachers (Zeichner, 1983; Rudduck, 1985; Kansanen, 2006; Ericson, 2008; EC, 2009). The curricula of these programs should cover all blocks of content and process concepts identified in the present analyses. Moreover, our findings should be compared against established curricular models of computer science education in Europe and the United States.

The methodological approach taken is important in efforts to consolidate curricular models of computer science education, as have been initiated by the Bologna process (European Ministers of Higher Education, 1999; 2005; 2006; Bologna Working Group on Qualifications Frameworks, 2005) in Europe (Mulder & van Weert, 2000; 2001) and by the organizations ACM (Association for Computing Machinery), AIS (Association for Information Systems), and IEEE-CS (Institute of Electrical and Electronic Engineers—Computer Society) (ACM, 2004a; 2004b; 2005a; 2005b; Gorgone et al., 2002) in the United States. The approach chosen in this study provides a way of generalizing curricular models of computer science that is based on methods which are well established in cross-cultural research.

In future research from the cross-cultural perspective, external validation studies (van der Vijver & Leung, 1997) should examine which contextual variables explain the differences found between German and US professors of computer science in the present study. Further empirical studies addressing more specific didactic issues are also warranted in this context. For example, empirical analyses should examine which central concepts (content and process concepts) should be covered at specific grade levels, which central concepts are appropriate for elementary instruction, and which instructional methods and which forms of social interaction should be used to teach central concepts.

Acknowledgements: The authors would like to thank Leonie Löw and Linda Hinderer for their assistance in conducting the questionnaire survey and Susannah Goss for final editing. The project was funded by the University of Education Ludwigsburg.

6. References:

Armoni, M., & Ginat, D. (2008). Reversing: A fundamental idea in computer science. *Computer Science Education, 18*(3), 213–230. <http://dx.doi.org/10.1080/08993400802332670>

Association for Computing Machinery (ACM). (2003). *A model curriculum for K-12 ACM computer science*. New York: ACM.

ACM. (2004a). *Curriculum guidelines for undergraduate degree programs in computer engineering*. New York: The Joint Task Force on Computing Curricula.

ACM. (2004b). *Curriculum guidelines for undergraduate degree programs in software engineering*. New York: The Joint Task Force on Computing Curricula.

ACM. (2005a). *Computing curricula 2005: The overview report*. New York: The Association for Computing Machinery (ACM), Association for Information Systems (AIS), The Computer Society (IEEE-CS).

ACM. (2005b). *Computing curricula information technology volume*. New York: SIGITE Curriculum Committee, Association for Computing Machinery.

ACM. (2008). *Computer science curriculum 2008*. New York: ACM.

Bauer, F. L. (2009). *Die Lage der Informatik in der Bundesrepublik Deutschland* [In German]. Berlin: Springer.

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching 20*(1), 45–73, 2001.

Berry, J. W., Poortinga, Y. H., Segal, M. H., & Dasen, P. R. (2002). *Cross-cultural psychology*. Cambridge, UK: University Press.

Bologna Working Group on Qualifications Frameworks. (2005). *Qualifications: A framework for qualifications of the European Higher Education Area.* Copenhagen, Denmark: Ministry of Science, Technology and Innovation.

Bruner, J. S. (1960). *The process of education*. Cambridge: Harvard University Press.

Burnell, L. J., Priest, J. W., & Durrett, J. R. (2002). Distributed multidisciplinary software development. *IEEE Software, 19*(5), 86–93. <http://dx.doi.org/10.1109/MS.2002.1032859>

Costa, A. L., & Liebmann, R. M. (1997d). Toward renaissance curriculum. In L. A. Costa & R. M. Liebmann (Eds.*), Envisioning process as content. Toward a renaissance curriculum* (pp. 1–20). Thousand Oaks, CA: Corwin Press.

Costa, A. L., & Liebmann, R. M. (Eds.). (1997a). *Envisioning process as content. Toward a renaissance curriculum*. Thousand Oaks, CA: Corwin Press.

Costa, A. L., & Liebmann, R. M. (Eds.). (1997b). *The process-centered school. Sustaining a renaissance community.* Thousand Oaks, CA: Corwin Press.

Costa, A. L., & Liebmann, R. M. (Eds.). (1997c). *Supporting the spirit of learning. When process is content.* Thousand Oaks, CA: Corwin Press.

Denning, P. J. (2003). Great principles of computing. *Communications of the ACM, 46*(11), 15–20. <http://dx.doi.org/10.1145/948383.948400>

Dillman, D. A. (2000). *Mail and Internet surveys: The tailored design method.* New York: Wiley.

Douglas, S., Farley, A., Lo, G., Proskurowski, A., & Young, M. (2010). Internationalization of computer science education. In *SIGCSE '10* (pp. 411–415), March 10–13, Milwaukee, WI. New York: SIGCSE.

Egea, K., Kim, S. K., Andrews, T., & Behrens, K. (2010). Approaches used by cross-cultural and cross-discipline students in teamwork for a first-year course in web design. In *Proceedings of the 12th Australasian Computing Education Conference* (ACE 2010) (pp. 87–96), Brisbane, Australia.

Emmott, S. (2006). *Towards 2020 science*. Cambridge, UK: Microsoft.

Ericson, B. (2008). *Ensuring exemplary teaching in an essential discipline. Addressing the crisis in computer science teacher certification*. New York: ACM.

European Commission (EC). (2009). *Common European principles for teacher competences and qualifications*. Retrieved August 22, 2011, from http://ec.europa.eu/education/policies/2010/doc/principles_en.pdf

European Ministers of Higher Education. (1999). *Bologna declaration of 19 June 1999. Bologna, Italy: The Secretariat of the Bologna Follow-Up Group*. Retrieved September 20, 2011 from http://www.bologna-bergen2005.no/Docs/00-Main_doc/990719BOLOGNA_DECLARATION.PDF

European Ministers Responsible for Higher Education. (2003). *Realising the European Higher Education Area. Berlin, Germany: The Secretariat of the Bologna Follow-Up Group*. Retrieved September 20, 2011, from www.bologna-bergen2005.no/Docs/00-Main_doc/030919Berlin_Communique.PDF

European Ministers Responsible for Higher Education. (2005). *European Higher Education Area: Achieving the goals. Bergen, Norway: The Secretariat of the Bologna Follow-Up Group*. Retrieved September 20, 2011, from http://www.bologna-bergen2005.no/Docs/00-Main_doc/050520_Bergen_Communique.pdf

Everitt, B. S., Landau, S., & Leese, M. (2001). *Cluster analysis*. London: Arnold.

Fuchs, C., & Hofkirchner, W. (2003). *Studienbuch Informatik und Gesellschaft*. Norderstedt: Books on Demand.

Gorgone, J. T., Davis, G. B., Valacich, J. S., Topi, H., Feinstein, D. L., & Longenecker, H. E. Jr. (2002). *Model curriculum and guidelines for undergraduate degree programs in information system*s. Atlanta, GA: Association for Information Systems.

Grinstein, G., Trutschl, M., & Cvek, U. (2001). High-Dimensional Visualizations. In *Data mining conference KDD workshop 2001* (pp. 7–19). New York: ACM Press.

Hadjerrouit, S. (2005a). Constructivism as guiding philosophy for software engineering education. *ACM SIGCSE Bulletin, 37*(4), 45–49. <http://dx.doi.org/10.1145/1113847.1113875>

Hadjerrouit, S. (2005b). Learner-centered Web-based instruction in software engineering. *IEEE Transactions on Education, 48*(1), 99–104. <http://dx.doi.org/10.1109/TE.2004.832871>

Harkness, J. (2003). Questionnaire translation. In J. Harkness, F. J. R. van de Vijver, & P. Mohler (Eds.), *Cross-cultural survey methods* (pp. 35–56). New York: Wiley.

Harkness, J., van de Vijver, F. J . R., & Mohler, P. (2003). *Cross-cultural survey methods*. New York: Wiley.

Hubert, L .J., & Levin, J. R. (1976). A general statistical framework for assessing categorical clustering in free recall. *Psychological Bulletin, 83*, 1072–1080. <http://dx.doi.org/10.1037/0033-2909.83.6.1072>

Huynh, H., & Feldt, L. S. (1976) Estimation of the Box correction for degrees of freedom from sample data in randomised block and split-plot designs. *Journal of Educational Statistics, 1*, 69–82. <http://dx.doi.org/10.2307/1164736>

Johnson, D. G., & Miller, K. W. (2009). Ethical issues for computer scientists. In A. B. Tucker (Ed.), *Computer science handbook* (2nd edition) (pp. 19–29). Boca Raton, FL: Chapman & Hall.

Kansanen, P. 2006. Constructing a research-based program in teacher education. In F. Oser, F. Achtenhagen, & U. Renold (Eds.), *Competence oriented teacher training* (pp. 11–22). Rotterdam, The Netherlands: Sense Publishers.

Kirk, E. (1994). *Experimental design*. Belmont, CA: Wadsworth.

Lerner, R. M., Easterbrooks, M.A., Mistry, J., & Weiner, B. (2003). *Handbook of psychology, Vol. 6: Developmental psychology*. New York: Wiley

Loidl, S., Mühlbacher, J., & Schauer, H. (2005). Preparatory knowledge: Propaedeutic in informatics. In R. T. Mittermeir (Ed.), *From computer literacy to informatics fundamentals* (pp. 105–115). New York: Springer. <http://dx.doi.org/10.1007/978-3-540-31958-0_14>

Machaniak, P. (2007). A social construction approach to computer science education. *Computer Science Education, 7*(1), 1–20. <http://dx.doi.org/10.1080/08993400600971067>

Moreno, L., González, C., Castilla, I., González, E. J., & Sigut, I. (2007). Use of constructivism and collaborative teaching in an ILP processors course. *IEEE Transactions on Education, 50*(2), 101–111.

<http://dx.doi.org/10.1109/TE.2006.886461>

Mueller, K. E., & Barton, C. N. (1989). Approximate power for repeated-measures ANOVA lacking sphericity. *Journal of the American Statistical Association, 84*(406), 549–555.

Mueller, K. E., LaVange, L. E., Ramey, S. L., & Ramey, C. T. (1992). Power calculations for general linear multivariate models including repeated measures applications. *Journal of the American Statistical Association, 87*(420), 1209–1226.

Mulder, F., & van Weert, T. (2000). *IFIP/UNESCO's informatics curriculum framework 2000 for higher education*. Paris, France: UNESCO.

Mulder, F., & van Weert, T. (2001). IFIP/UNESCO's informatics curriculum framework 2000 for higher education. *Inroads SIGCSE Bulletin, 33*(4), 75–83. <http://dx.doi.org/10.1145/572139.572177>

Nievergelt, J. (1980). Computer science education: An emerging consensus on basic concepts. In S. H. Lavington (Ed.), *Information Processing 80* (pp. 927–933). Amsterdam, The Netherlands: North Holland.

Nievergelt, J. (1990). Computer science for teachers: A quest for classics and how to present them. In D.H. Norrie & H.W. Six (Ed.), *Computer assisted learning, lecture notes in computer science 438* (pp. 2–15). New York: Springer.

Parker, J. C., & Rubin, L. J. (1966). *Process as content. Curriculum design and the application of knowledge*. Chicago, IL: Rand McNally.

Rudduck, J. (1985). Teacher research and research-based teacher education. *Journal of Education for Teaching, 11*(3), 281–289. <http://dx.doi.org/10.1080/0260747850110305>

Saraswathi, T. S. (2003). *Cross-cultural perspectives in human development: Theory, research and applications*. New York: Sage.

Schwill, A. (1994). Fundamental ideas of computer science. *EATCS Bulletin, 53*, 274–295.

Scime, A. (2008). Globalized computing education: Europe and the United States. *Computer Science Education, 18*(1), 43–64. <http://dx.doi.org/10.1080/08993400701869491>

Shapiro, E., Harel, D., Bishop, C., & Muggleton, S. (2006). The fundamental role of computer science concepts in science. In S. Emmott (Ed.), *Towards 2020 science* (pp. 24–25). Redmond, CA: Microsoft.

Tucker, A. B., & Wegner, P. (2009). The discipline and its impact. In A. B. Tucker (Ed.), *Computer science handbook* (2nd edition) (pp. 3–18). Boca Raton, FL: Chapman & Hall.

van de Vijver, F., & Leung, K. (1997). *Methods and data analysis for cross-cultural research*. Thousand Oaks, CA: Sage.

Vaux, A., & Briggs, C. S. (2005). Conducting mail and Internet surveys. In F.T.L. Leong, & J.T. Austin, *The psychology research handbook* (pp. 186–209). Thousand Oaks, CA: Sage.

Ward, J. H. (1963). Hierarchical groupings to optimize an objective function. *Journal of the American Statistical Association, 58*, 236–244.

Weinberg, J. B., White, W. W., Karacal, C., Engel, G., & Hu, A. (2005). Multidisciplinary teamwork in a robotics course. In *Proceedings of the 36th SIGCSE Technical Symposium on computer science education*, St. Louis, MI, pp. 446–450. New York: ACM Press.

Winer, B. J., Brown, D. R., & Michels, K. M. (1991). *Statistical principles in experimental design*. Boston, MA: McGraw-Hill.

Wursthorn, B. (2005). Fundamental concepts of computer science in a Logo-environment. In G. Gregorczyk, A. Walat, W. Kranas, & M. Borowiecki (Eds.), *Digital tools for lifelong learning. Proceedings of the tenth European Logo Conference* (pp. 219–227). Warsaw, Poland: Centre for Informatics and Technology in Education.

Zeichner, K. M. (1983). Alternative paradigms of teacher education. *Journal of Teacher Education, 34*(3), 3–9. <http://dx.doi.org/10.1177/002248718303400302>

Zendler, A., & Spannagel, C. (2008). Empirical foundation of central concepts for computer science education. *ACM Journal on Educational Resources in Computing, 8*(2), Article No. 6.

Zendler, A., Spannagel, C., & Klaudt, D. (2008). Process as content in computer science education: Empirical determination of central processes. *Computer Science Education, 18*(4), 231–245.

Zendler, A., Spannagel, C., & Klaudt, D. (2011). Marrying content and process in computer science education. *IEEE Transactions on Education, 54*(3), 387–397.

Appendix

Central content and process concepts

Questionnaire

Each cell represents a combination of a concept and a process and requires an integer from 0 (no importance) to 5 (great importance) indicating the relevance of the combination. *Fill in each cell on a scale from 0 to 5 (integers only).*

no importance 0 1 2 3 4 5 great importance

	***Process* No.**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
No.	***Concept***	*analyzing*	*categorizing*	*classifying*	*collaborating*	*communicating*	*comparing*	*creating and inventing*	*finding cause-and-effect relationships*	*finding relationships*	*generalizing*	*investigating*	*ordering*	*presenting*	*problem solving and problem posing*	*questioning*	*transferring*
1	algorithm																
2	communication																
3	computation																
4	computer																
5	data																
6	information																
7	language																
8	model																
9	problem																
10	process																
11	program																
12	software																
13	structure																
14	system																
15	test																

It is important that you give 16 ratings for each row. At the end of your review, you should have a filled 15 ×16 matrix.

Groups

a_1 = German professors of computer science
a_2 = US professors of computer science

Groups	**Content concepts** \ **Process concepts**	c_1 = analyzing	c_2 = categorizing	c_3 = classiyfing	c_4 = collaborating	c_5 = communicating	c_6 = comparing	c_7 = creating and inventing	c_8 = finding cause-and-effect r.	c_9 = finding relationships	c_{10} = generalizing	c_{11} = investigating	c_{12} = ordering	c_{13} = presenting	c_{14} = problem solving and posing	c_{15} = questioning	c_{16} = transferring	*Grand means*
a_1	b_1 = algorithm	4.46	3.04	3.42	1.42	1.96	3.46	4.13	2.67	2.54	3.83	3.58	1.67	2.46	4.25	2.38	3.33	3.04
	b_2 = communication	3.38	2.29	2.42	3.50	4.13	2.08	2.25	2.58	2.63	2.17	2.71	1.38	2.33	2.25	2.63	2.13	2.55
	b_3 = computation	3.79	2.67	3.00	2.13	2.08	2.75	2.67	2.71	2.92	3.04	3.08	1.58	2.42	3.71	2.46	2.46	2.72
	b_4 = computer	2.83	2.92	2.58	2.83	2.88	2.21	2.50	1.96	1.67	1.50	2.21	1.04	1.79	2.25	2.29	1.96	2.21
	b_5 = data	4.00	3.75	3.83	1.54	2.17	3.25	1.50	2.79	3.71	2.92	2.83	3.71	3.42	2.50	2.75	2.33	2.94
	b_6 = information	4.29	3.79	4.21	1.96	2.58	3.25	1.83	2.50	3.79	3.42	2.83	3.33	3.54	2.63	3.00	2.92	3.12
	b_7 = language	3.46	3.08	2.92	2.17	2.88	2.33	2.67	1.75	2.67	2.71	2.42	1.33	2.08	2.83	1.88	2.13	2.46
	b_8 = model	4.17	3.42	3.38	2.46	2.38	3.13	3.58	2.75	3.46	4.13	3.29	1.88	2.58	3.46	2.63	3.13	3.11
	b_9 = problem	4.54	3.63	3.83	2.58	2.50	2.88	2.04	2.88	3.08	3.17	4.00	1.75	2.50	4.50	3.54	2.83	3.14
	b_{10} = process	4.13	3.04	2.96	2.67	3.00	2.42	2.00	2.92	2.71	2.42	3.04	1.71	2.13	2.63	2.38	2.33	2.65
	b_{11} = program	4.08	2.38	2.63	2.42	2.38	2.13	3.38	2.33	2.21	2.50	2.71	1.42	2.25	3.50	2.04	2.75	2.57
	b_{12} = software	4.04	2.71	2.79	3.00	2.79	2.08	3.67	2.38	2.29	2.50	2.67	1.13	2.17	3.17	2.25	2.46	2.63
	b_{13} = structure	4.29	3.08	3.33	1.42	1.54	2.42	2.67	2.75	3.50	3.79	3.13	2.17	2.88	2.33	2.58	2.71	2.79
	b_{14} = system	4.25	2.75	2.79	2.71	2.46	2.04	2.83	2.63	2.92	2.79	3.17	1.38	2.79	3.17	2.46	2.46	2.72
	b_{15} = test	3.29	2.63	2.38	1.96	1.83	2.88	3.00	3.63	2.79	1.92	3.04	1.42	2.17	2.71	3.21	2.08	2.56
	Grand means	3.93	3.01	3.10	2.32	2.50	2.62	2.71	2.61	2.86	2.85	2.98	1.79	2.50	3.06	2.56	2.53	*2.75*
a_2	b_1 = algorithm	4.80	3.07	3.00	1.40	2.80	3.60	4.20	2.27	3.13	4.00	3.27	1.80	2.60	3.93	3.10	3.67	3.17
	b_2 = communication	2.57	2.30	2.13	3.77	4.27	2.13	2.67	2.73	2.40	2.40	2.60	1.80	2.80	2.80	2.87	2.20	2.65
	b_3 = computation	3.80	3.00	2.87	2.13	2.47	3.00	3.20	3.07	3.10	3.37	3.47	2.63	2.70	3.80	2.87	2.73	3.01
	b_4 = computer	3.40	2.43	2.57	2.20	2.20	2.67	3.33	2.33	2.33	2.67	3.20	2.43	2.13	3.20	3.07	2.53	2.67
	b_5 = data	4.40	3.93	4.00	2.23	2.53	3.80	2.13	3.47	4.33	3.33	2.73	3.07	3.27	2.90	3.60	2.80	3.28
	b_6 = information	4.40	3.73	4.00	2.47	3.80	3.60	2.80	3.33	3.93	3.00	3.40	3.33	3.80	3.13	3.80	3.20	3.48
	b_7 = language	2.57	2.67	2.67	2.80	3.90	2.37	2.53	2.13	2.50	2.73	2.60	2.00	2.83	2.80	2.63	2.93	2.67
	b_8 = model	4.87	3.67	3.33	2.00	2.93	3.47	3.53	2.87	3.27	4.47	3.60	2.07	2.80	3.07	3.33	3.33	3.29
	b_9 = problem	4.33	3.67	3.73	2.47	3.07	2.73	2.90	3.07	3.07	3.67	3.80	2.17	3.53	4.47	3.13	2.77	3.29
	b_{10} = process	4.07	2.77	2.53	2.87	2.80	2.27	2.63	3.23	2.73	2.87	2.90	2.67	2.47	2.93	2.57	2.73	2.82
	b_{11} = program	4.33	2.33	2.47	3.07	2.80	2.80	4.40	4.00	3.47	3.17	2.97	2.87	2.80	3.73	2.47	3.27	3.18
	b_{12} = software	4.20	2.33	2.57	3.53	3.00	2.93	4.20	4.13	3.03	3.07	2.90	2.47	2.80	3.80	2.53	2.87	3.15
	b_{13} = structure	3.83	3.47	3.40	1.67	2.30	2.80	2.80	2.73	3.27	3.13	2.40	2.27	2.20	2.80	2.40	2.80	2.77
	b_{14} = system	4.00	2.80	3.00	2.60	2.73	2.87	3.47	3.20	2.50	2.67	2.93	1.73	2.40	3.33	2.93	2.33	2.84
	b_{15} = test	3.53	2.33	2.77	1.93	2.27	2.73	2.73	4.10	2.83	3.07	2.93	2.33	2.40	3.67	2.87	2.10	2.79
	Grand means	3.94	2.97	3.00	2.48	2.92	2.92	3.17	3.11	3.06	3.17	3.05	2.38	2.77	3.36	2.94	2.82	*3.00*

Figure A-1. Means for the 2•15×16 split-plot design (n_1=24; n_2=15)

Process concepts / **Content concepts**	c_1 = analyzing	c_2 = categorizing	c_3 = classiyfing	c_4 = collaborating	c_5 = communicating	c_6 = comparing	c_7 = creating and inventing	c_8 = finding cause-and-effect r.	c_9 = finding relationships	c_{10} = generalizing	c_{11} = investigating	c_{12} = ordering	c_{13} = presenting	c_{14} = problem solving and posing	c_{15} = questioning	c_{16} = transferring	*'Grand means*
b_1 = algorithm	4.59	3.05	3.26	1.41	2.28	3.51	4.15	2.51	2.77	3.90	3.46	1.72	2.51	4.13	2.65	3.46	3.09
b_2 = communication	3.06	2.29	2.31	3.60	4.18	2.10	2.41	2.64	2.54	2.26	2.67	1.54	2.51	2.46	2.72	2.15	2.59
b_3 = computation	3.79	2.79	2.95	2.13	2.23	2.85	2.87	2.85	2.99	3.17	3.23	1.99	2.53	3.74	2.62	2.56	2.83
b_4 = computer	3.05	2.73	2.58	2.59	2.62	2.38	2.82	2.10	1.92	2.67	2.59	2.43	1.92	2.62	2.59	2.18	2.49
b_5 = data	4.15	3.82	3.90	1.81	2.31	3.46	1.74	3.05	3.95	3.08	2.79	3.46	3.36	2.65	3.08	2.51	3.07
b_6 = information	4.33	3.77	4.13	2.15	3.80	3.38	2.21	2.82	3.85	3.26	3.05	3.33	3.64	2.82	3.31	3.03	3.31
b_7 = language	3.12	2.92	2.82	2.41	3.27	2.35	2.62	1.90	2.60	2.72	2.49	1.59	2.37	2.82	2.17	2.44	2.54
b_8 = model	4.44	3.51	3.36	2.28	2.59	3.26	3.56	2.79	3.38	4.26	3.41	1.95	2.67	3.31	2.90	3.21	3.18
b_9 = problem	4.46	3.64	3.79	2.54	2.72	2.82	2.37	2.95	3.08	3.36	3.92	1.91	2.90	4.49	3.38	2.81	3.20
b_{10} = process	4.10	2.94	2.79	2.74	2.92	2.36	2.24	3.04	2.72	2.59	2.99	2.08	2.26	2.74	2.45	2.49	2.72
b_{11} = program	4.18	2.36	2.56	2.67	2.54	2.38	3.77	4.00	3.47	2.76	2.81	2.87	2.46	3.59	2.21	2.95	2.97
b_{12} = software	4.10	2.56	2.71	3.21	2.87	2.41	3.87	4.13	2.58	2.72	2.76	2.47	2.41	3.41	2.36	2.62	2.95
b_{13} = structure	4.12	3.23	3.36	1.51	1.83	2.56	2.72	2.74	3.41	3.54	2.85	2.21	2.62	2.51	2.51	2.74	2.78
b_{14} = system	4.15	2.77	2.87	2.67	2.56	2.36	3.08	2.85	2.76	2.74	3.08	1.51	2.64	3.23	2.64	2.41	2.77
b_{15} = test	3.38	2.51	2.53	1.95	2.00	2.82	2.90	3.81	2.81	3.07	3.00	1.77	2.26	3.08	3.08	2.09	2.69
Grand means	3.93	2.99	3.06	2.38	2.71	2.73	2.89	2.95	2.99	3.07	3.01	2.19	2.60	3.17	2.71	2.64	*2.88*

Figure A-2. Weighted means of the German and US professors of computer science (*N*=39)—concepts in alphabetical order