A Cohesion Metric for the Definition of Activities in a Workflow Process

H.A. Reijers
Department of Technology Management, Eindhoven University of Technology,
PO Box 513, NL-5600 MB, Eindhoven, The Netherlands
e-mail: h.a.reijers@tm.tue.nl

Abstract

This paper introduces a cohesion metric that can be used for the identification of weakly cohesive activities in a workflow design. Also, a heuristic is presented that is based on this cohesion metric to decide between various workflow design alternatives. Both a theoretical and an empirical evaluation give a positive support its soundness. The inspiration for the introduced notion is derived from cohesion metrics in software engineering.

1. INTRODUCTION

Workflow management projects typically start with the design of a business process as a network of related activities. After the design phase, a formal representation of that process can be used to configure a workflow management system. One of the functions of such a system is that it can allocate activity instances to the workers in that process at run-time (Jablonski and Bussler, 1996; Van der Aalst & Van Hee, 2002). However central the *activity* concept may be within such a setting, it is the author's experience in various workflow projects (De Crom and Reijers, 2001; Reijers, 2002.) that the knowledge of identifying and delineating activities within a business process is limited and can result in ill-defined activities.

The results of ill-defined activities on the operational performance of a process may be substantial. One may think of the example of activities that are needlessly small. This increases the number of hand-offs between activities, with a corresponding increase of errors (Seidmann and Sundararajan, 1997). Activities that are too large may cause inflexibility within a business process, since its underlying operations must be performed regardless of their merits under the circumstances (Van der Aalst, 2000).

The aim of this paper is to provide some tangible guidance for activity definition in the form of a heuristic. It is meant to make the well-known notion of a 'logical unit of work' operational. The application area is the design of workflow processes. The heuristic we propose is based upon a cohesion metric for activities, as inspired by similar notions in software engineering.

The structure of this paper is as follows. First, we will introduce some basic concepts and give a short overview of existing activity definition heuristics in the workflow management field. Next we will present the cohesion metric we mentioned earlier, as well as a heuristic for its use. After its introduction and the presentation of some examples, we will subject the cohesion metric to both a theoretical and empirical evaluation. We end this paper with some concluding remarks and directions for further research.

2. ACTIVITY DESIGN IN WORKFLOW MANAGEMENT

2.1. Terminology

An *activity* is a specification of a part of work to be accomplished. We use '*workflow process*' as a synonym for a specific type of business process, which itself is a conceptual way of organizing work and resources by distinguishing a set of related activities. Workflow processes are usually found in administrative contexts (e.g. banking, insurance, government, etc.). They are particularly suitable to be supported by workflow management systems.

Each single activity that is distinguished within a workflow process may be divided into a number of operations. Operations are used to identify small parts of work in a way that is still useful within the business context. In general, it is also possible to distinguish an activity without mentioning the operations it comprises (non-determinism).

We interpret the matter of activity definition as the formulation of a goal and/or the assignment of operations to an activity within the context of a single workflow process. Part of the work in defining activities involves an evaluation of its properties, such as its size, its workability, its performance, etc. Although a broader view on an activity definition may also include matters such as the development of work instructions, various views and abstraction levels, supporting information systems, interfaces etc., these are outside the scope of this paper.

In choosing the respective terminology, we aspired consistency with the standards of the Workflow Management Coalition (Fischer, 2001).

2.2. Workflow Management

When taking the perspective of an organization, it immediately becomes clear that there are many factors that determine - or at least influence - how activities should be defined. One can think of regulations and considerations that emerge from human resource management, ergonomics, quality management, social sciences, accountancy and various other fields.

One specific view is that of workflow management (WfM), which produces some very pragmatic directions on activity definition. Sharp and McDermott (2001) illustrate this by allowing on each of the three to five hierarchic levels of a workflow process a number of five to seven activities. In other words, no matter what the operations are, they should be fitted in somewhere.

Van der Aalst and Van Hee (2002) note that to prevent problems in supporting processes by WfM systems it is necessary to only regard as an activity a logical unit of work (LUW). This means that the so-called *ACID* properties known from transaction processing apply: atomicity, consistency, isolation, and durability (Harder and Reuter, 1983). The authors also state their decomposition criterion for distinguishing a LUW: There must exist *unity of time, place, and operation*.

The unity of time, place, and operation criterion often does not act as an imperative, but rather as a first step for activity decomposition, according to Van der Aalst and Van Hee (2002). They give the following additional decomposition criteria:

- the *recognizability of an activity* by the members of the organization that must perform it is important, with a clear function and objective.
- *sensible interim states*: all resulting interim states in the process caused by activity decomposition should be sensible.
- *acceptable "commit work" for each of the process activities*: violations of the ACID properties should be acceptable, especially with respect to possible rollback and the split up of tasks.

Van der Aalst and Van Hee admit that in practice it is not easy to address the ACID properties, because of the properties and limitations of current workflow systems. A partial solution to this problem is given by Grefen et al. (2001), who distinguish high-level (long-living) and low-level (relatively short-living) processes. The latter are subprocesses of the former, but both have different requirements. The low-level requires *strict* execution of the ACID properties. The high-level needs relaxation of the atomicity and isolation requirements. Their WIDE model for WfMS's supports this view (Grefen et al., 1999).

In conclusion, in the WfM field heuristics and rules of thumb are used to identify operations that more or less naturally belong together. Characteristically, decisions on splitting up or combining activities are context-sensitive. All the given rules provide a considerable degree of design freedom.

3. A COHESION METRIC FOR WORKFLOW ACTIVITIES

3.1. Introduction

In software engineering, manipulations (declarations, assignments, invocations, etc.) that are strongly related are preferably grouped together within the same module or class (Stevens et al., 1974). There are clear implications for the maintainability and re-usability of programs by using this approach and there is also considerable empirical evidence that the resulting computer programs contain fewer errors (Card et al., 1986; Selby and Basili, 1991).

Workflow processes are in some sense similar to computer programs, as they primarily involve *information processing steps*: checks, decisions, computations, copying, etc, Also, in workflow processes the administration of information is often vital (customer data, order information, etc.). Furthermore, workflow processes consist of activities, where computer programs are divided into modules or classes. Also, where modules contain statements and classes methods, activities consist of operations. For more information on the characterization of workflow processes, we refer the reader to Reijers (2002).

Because of the advantages of cohesion metrics in software design and the similarities between programs and workflow processes, it seems both fruitful and plausible to pursue an analogon to the cohesion metric for the sake of activity definition in workflow processes.

3.2. The Activity Cohesion Metric

Prior to the formulation of the cohesion metric and the presentation of some examples, we introduce its supporting notions. For the sake of clarity, we do not consider an entire workflow process. Instead we directly zoom in on a part of the process for which it is unclear how to define activities, a so-called *operations structure*. We assume as given a sense of *operations* that take place within this part. Operations in our view can be seen as information functions, which take as *inputs* zero or more pieces of information and produce one new piece of information, its *output*. This

view on the elementary operations within a workflow process is consistent with the workflow design methodology Product-based Workflow Design (Van der Aalst et al., 2001; Reijers, 2002) and has proven to be a practically applicable metaphor in several cases (De Crom and Reijers, 2001; Reijers, 2002).

Definition 1. (Operations Structure). An *operations structure* is a tuple (D, O) with:
- D: the set of *information elements* that are being processed,
- $O = \{ (p, cs) \in D \times \Pi(D) \}$ is a set of *operations* on the information elements, such that there are no 'dangling' information elements and no value of an information element depends on itself:
  - $R = \{(p, c) \in D \times D \mid \exists (p, cs) \in O : c \in cs \}$ is connected and acyclic.

The job of process design is to impose on an operations structure activities that partition the set of operations.

Definition 2. (Valid activity). Given an operations structure (D, O), any subset $t \subseteq O$ is a *valid activity* on the operations structure, or simply a *activity*.

Definition 3. (Valid activity ordering). Given an operations structure (D, O), the tuple (T, F) is a valid activity ordering on that operations structure iff:
- T is a set of valid activities, $T \subseteq \Pi(O)$, such that:
  1. $\forall o \in O : (\exists t \in T : o \in t)$.
- F is a partial ordering on F, $F \subseteq T \times T$, such that for each $t \in T$:
  2. $\forall t, u \in T : \left((\exists (p, cs) \in t, (q, ds) \in u : q \in cs) \Rightarrow (u, t) \in F^* \right)$.

Within this definition it is expressed by 1. that all operations from the operation structure should appear at least once in one activity. Condition 2. enforces that when one operation depends on the output of another, then the respective tasks they are part of are ordered such that they respect this dependency.

The definition of our cohesion metric, then, depends on two important parts: the *relation cohesion* and the *information cohesion*. The relation cohesion gives a measure on how much the different operations within one activity are related.

Definition 4. (Activity relation cohesion). For a valid activity t on an operation structure (D, O), its relation cohesion $\lambda(t)$ is defined as follows:

$$l(t) = \begin{cases} \dfrac{\sum_{(p,cs) \in t} \left| \{ (q, ds) \in t \setminus \{(p, cs)\} \mid (\{p\} \cup cs) \cap (\{q\} \cup ds) \neq \emptyset \} \right|}{|t| \cdot |t - 1|}, \text{for } |t| > 1 \\ 0, \text{for } |t| \leq 1. \end{cases}$$

The other component of our cohesion metric, the activity information cohesion, focuses on the sharing of information elements.

Definition 5. (Activity information cohesion). For a valid activity t on an operation structure (D, O), its information cohesion $\mu(t)$ is defined as follows:

$$m(t) = \begin{cases} \dfrac{\left| \{ d \in D \mid \exists (p, cs), (q, ds) \in t : d \in (\{p\} \cup cs) \cap (\{q\} \cup ds) \wedge (p, cs) \neq (q, ds) \} \right|}{\left| \{ d \in D \mid \exists (p, cs) \in t : d \in (\{p\} \cup cs) \} \right|}, \text{for } |t| > 0 \\ 0, \text{ for } |t| = 0 \end{cases}$$

The relation cohesion and information cohesion coefficients may appear to be related. Nonetheless, it is possible to think of an activity with one type of cohesion that equals 1, but another type that is almost 0. Because of limitations on space, actual examples are not given here.

The total cohesion of an activity is now given as the product of both the relation and information cohesion. An activity has to score high on both cohesion metrics to say it is cohesive in total. Clearly, an extreme score on one coefficient may outweigh a mediocre result on the other, which is seen as satisfactory.

Definition 6. (Activity cohesion). For a valid activity t on an operation structure (D, O), its (general) cohesion $c(t)$ is defined as follows:

$$c(t) = \lambda(t) \cdot \mu(t)$$

3.3. Application of the Cohesion Metric

Let us assume that there is an activity X, which is relatively incohesive in an overall workflow design on the basis of the presented cohesion metric. It is subsequently considered to be split up into validly ordered activities A and B. An evaluation that could take place on the basis of the same activity cohesion is then as follows:

1. Determine the cohesion of A and B (the cohesion of X is already known),
2. If both cohesion coefficients of A and B are higher than that of X, then the division into A and B is preferable,
3. If the cohesion coefficient of X is higher than both cohesion coefficients of A and B, then the larger activity X is preferable
4. In all other cases, the heuristic is indecisive.

Note that our heuristic *does not describe how the candidates A and B can be determined*. Obviously, in small enough cases it is feasible to generate a great number of partitions, but it grows exponentially in the number of operations that are considered. Also note that a similar approach could be taken when activity X is considered to be integrated with another activity Y, resulting in activity C.

Consider the example in Figure 1, based on the operations structure (D_1, O_1) with $D_1 = \{a, b, \ldots, p\}$ and $O_1 = \{(c, \{a, b\}), (f, \{d, e\}), (g, \{c, f\}), (h, \{g\}), (p, \{m, n\}), (l, \{h, i, j, k\}), (o, \{m, n\}), (p, \{o, l\})$. It represents two alternatives. Alternative X consists of one activity that comprises all these operations. The other alternative consists of a valid ordering of activities A and B. Note that an arrow leading from one information element to another signifies that the former is needed as an input for the other in some operation. Also note that the ordering of activities A and B is valid.

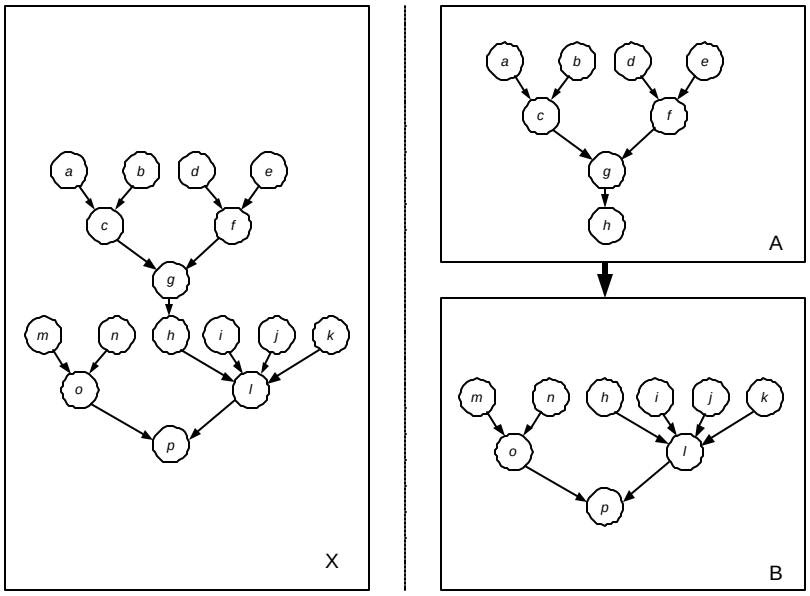

Figure 1: Example 1

If we consider operation $(c, \{a, b\})$ we see that it has a relation with just one other operation, namely $(g, \{c, f\})$. After all, output c of the former is an input of the latter. On average for activity X, each of its operations has a relation with 12/7 other relations, being the quotient of the summed number of relations over all operations and the total number of relations. The maximum number of relations that any operation within a set of 7 could have equals 6. Therefore, the relation cohesion of activity X equals 12/(7*6) = 2/7.

Furthermore, there are 6 information elements – c, f, g, h, o, and p – that are shared by the operations within X. The rest of the 16 elements is in use by only one operation. Therefore, the information cohesion of activity X equals 6/16. The total cohesion of activity X is the product of its relation and information cohesion: 2/7 * 6/16 =

12/112.Similarly we can compute the cohesion coefficients of activities A and B. The results of this exercise are given in Table 1.

	X	A	B
relation cohesion	2/7	5/12	2/3
information cohesion	6/16	3/8	2/9
total cohesion	12/112 (≈ 0.1)	15/96 (≈ 0.2)	4/27 (≈ 0.2)

Table 1: Cohesion Coefficients for Example 1

If we apply our heuristic to this example, then the division of the operations structure into A and B is preferable over the single activity X. This appeals to the intuition that activity X is divided into two parts that are only related to each other through operation $(g, \{h\})$. Somebody who is to perform this task may easily wonder what the processing of a, b, c, d, e, f has to do with the processing of m, n, i, j, k.

To appreciate the heuristic's opposite discrimination consider the example as given in Figure 2, based on the operations structure (D_2, O_2) with $D_2 = \{a, b,\ldots, i\}$ and $O_2 = \{(a, \{g, h\}), (d, \{i\}), (c, \{a, b\}), (e, \{b, d\}), (f, \{c, e\})\}$. It again represents two alternatives in the fashion of the first example. The various cohesion coefficients are given in Table 2.

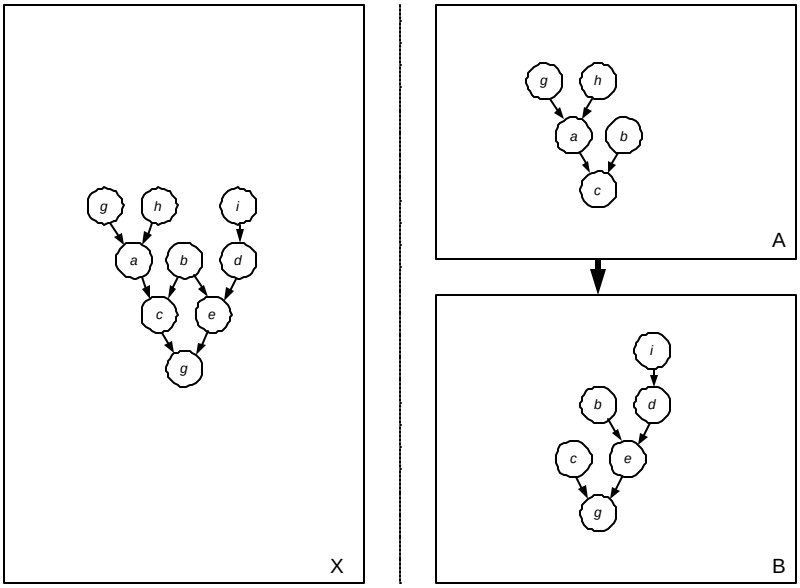

Figure 2: Example 2

	X	A	B
relation cohesion	1/2	1	2/3
information cohesion	5/9	1/5	1/3
total cohesion	5/18 (≈ 0.3)	1/5 (= 0.2)	2/9 (≈ 0.2)

Table 2: Cohesion Coefficients for Example 2

4. EVALUATION

4.1. Theoretical Perspective

One could wonder if the introduced cohesion metric has a sound theoretical basis. For the sake of comparison: In computing science many metrics exist to express the quality of some piece of software, but many of these metrics have been subject to wide criticism because of the lack of such a theoretical basis (e.g. Vessey and Weber, 1984).

To the author's knowledge, quality criteria are lacking so far in the area of workflow design. Therefore, we consider a set of principles as defined by Chidamber and Kemerer (1994), which in turn have been derived from Weyuker (1988). These principles specifically aim at providing support when specifying cohesion metrics in objected oriented software design. Although the differences between this area and that of process design are clear, we refer to it in lack of more specific support.

The principles of Chidamber and Kemerer (1994), are defined on classes, which in Object-Oriented Design (OOD) can be seen as abstractions of the problem space. They are the following:

1. *Noncoarseness*: Given a class P and a metric m, another class Q can always be found such that $m(P) \neq m(Q)$. This implies that not every class can have the same value for a metric, otherwise it has lost its value as a measurement.
2. *Nonuniqueness (Notion of Equivalence)*:There can exist distinct classes P and Q such that $m(P) = m(Q)$. This implies that two classes can have the same metric value, i.e., the two classes are equally complex.
3. *Design Details are Important*: Given two class designs, P and Q, which provide the same functionality, does not imply that $m(P) = m(Q)$. The specifics of the class must influence the metric value. The intuition behind property is that even though two class designs perform the same function, the details of the design matter in determining the metric for the class.
4. *Monotonicty*: For all classes P and Q, the following must hold $m(P) \leq m(P+Q)$ and $m(Q) \leq m(P+Q)$ where $P+Q$ implies combination of P and Q. This implies that the metric for the combination of two classes can never be less than the metric for either of the component classes.
5. *Nonequivalence of Interaction*: $\exists P, \exists Q, \exists R$, such that $m(P) = m(Q)$ does not imply that $m(P+R) = m(Q+R)$. This suggests that interaction between P and R can be different than interaction between Q and R resulting in different complexity values for $P + R$ and $Q + R$.
6. *Interaction increases Complexity*: $\exists P, \exists Q$, such that $m(P) + m(Q) < m(P+Q)$. The principle behind this property is that when two classes are combined, the interaction between classes can increase the complexity metric value.

We will now consider each of these requirements on our cohesion metric, interchanging the class notion for that of an activity. For this evaluation, we will assume that in general an operations structure will contain a non-trivial number of operations which are not totally unrelated (i.e., there are at least two operations that share at least one information element).

4.1.1. Noncoarseness

With respect to the first property, noncoarseness, suppose that there is an activity t defined on a certain operations structure. If $c(t)$ equals zero, it is always possible to add two related operations such that $c(t)$ increases. If $c(t)$ is unequal to zero, it is possible to take operations away from it until its cohesion equals zero. Therefore, noncoarseness is guaranteed on theoretical grounds. More practically, taking away or adding an operation will *mostly* affect both the relation cohesion and the information cohesion of the activity.

4.1.2. Nonuniqueness

If we consider the second property, nonuniqueness, it is immediately clear that each activity with only one operation – regardless which one - has a cohesion of 0. So on theoretical grounds this property is satisfied. Practically, it will often be possible to define two activities with the same cohesion if an operations structure is sufficiently large. For instance, it will then be possible to find two sets of two operations each such that both sets (activities) involve an equal number of information elements of which only one information element is shared among the two operations it consists of. Both these activities will exactly have the same cohesion.

4.1.3. Design Details Are Important

An evaluation of the third property, design details are important, requires us to evaluate the notion of functionality within the context of process design. We assume that two activities are functionally the same when they share the same outputs, i.e. they contain operations with equal outputs that are *not* used as inputs by other operations within these activities. In practical operations structures it will often be the case that there are two operations with the same output, but with different inputs, e.g. determining somebody's suitability for a job by an interview or by means of a

psychological test. The specific choice for one of the alternatives to include in an activity will very likely be of influence on its cohesion.

4.1.4. Monotonicty

The fourth property of monotonicity is in general *not* satisfied by our notion of cohesion. There is a very good explanation for this: The cohesion criterion is explicitly targeted to decide whether it is wise to combine activities or not. If cohesion would have been a monotonic property, combining activities *always results* in a higher cohesion. This would have made the criterion worthless for our purpose. The original thought behind this property may be inspired by such simple complexity metrics as 'number of lines in code' and 'number of methods in a class'. It is interesting to note that this property is neither satisfied by the suite of cohesion metrics that Chidamber and Kemerer (1994) themselves propose.

4.1.5. Nonequivalence of Interaction

It is easy to see that the fifth property, Nonequivalence of Interaction, will be satisfied in most practical cases. We only have to consider two different activities with equal cohesion and an operation that is related to one or more activities within the first, but to none of the activities in the other activity. In adding the operation to both activities, the cohesion may respectively increase in the first case and will *always* decrease in the second.

4.1.6. Interaction Increases Complexity

Finally, the sixth property, Interaction Increases Complexity, can be satisfied by choosing any two operations from an operations structure that share an input or output. Two separate activities with only one of these operations will have a cohesion that equals zero; one activity that includes both operations will have a positive cohesion, so that the property is satisfied. Although it will be not very common that $m(P) + m(Q) < m(P+Q)$ in a practical case – the cohesion differences are very large – our criterion is specifically intended to identify cases where $m(P) < m(P+Q)$ and $m(Q) < m(P+Q)$. After all, in this situation we would prefer to combine P and Q (see example 2).

4.1.7. Discussion

On the basis of the previous evaluation, the cohesion metric we presented could be said to satisfy all relevant theoretical requirements of Chidamber and Kemerer (1994). Obviously, there are many reservations to be made, for example the absence of more context-specific theoretical requirements. As this is really a best effort, it is interesting to take a look at an empirical evaluation of the heuristic as well.

5. EMPIRICAL PERSPECTIVE

5.1. Web-Based Survey

The approach we followed for the empirical evaluation is to test the heuristic by applying it to a set of design dilemmas and compare its outcomes to the judgment of human experts. For this purpose, we used a digital web-based survey, which contained ten design dilemma's in the same spirit as the examples of Figure 1 and Figure 2. A respondent must choose for each of the dilemma's on a three-point Likert scale whether he or she:

- prefers to combine the operations in one large activity,
- has no preference for combining or splitting up these activities, or
- supports the split-up of the same operations in the two *given* activities

The respondent is instructed to follow his intuition whether the operations as depicted seem to "belong together" or not. The only thing what is explained to the respondent are the meaning of the used symbols in each figure and the context of workflow design. A screen-shot of one of the presented dilemma's in the web-based survey is given in Figure 3.

Fifteen Dutch workflow designers, working for management consulting companies, one large bank, and one utilities provider, were asked to cooperate in this survey. Fourteen of them actually responded before the dead-line. The average number of workflow design projects they participated in ranged from two to twenty five, with an average of ten.

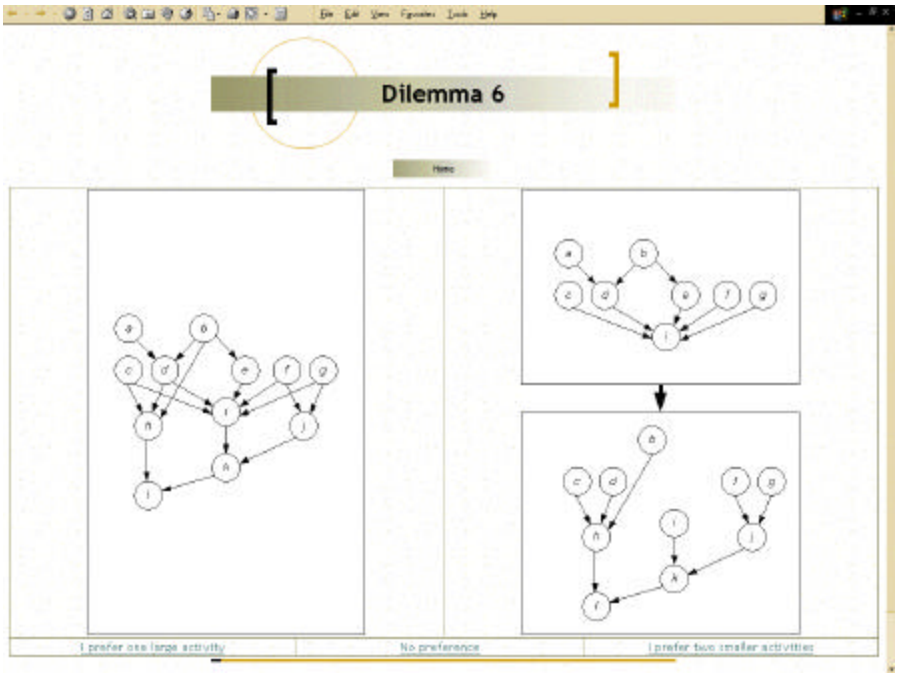

Figure 3: Screenshot of the Web-Survey

For the analysis of the results we used statistical methods for *rater agreement*, as often used in medical settings to compare expert's opinions on the same data. For more background on rater agreement statistics, see Uebersax (1992) and Uebersax (2001). We computed the Pearson correlation between the respondents' average score and the heuristic outcome. In Table 3, we have represented the outcomes of the web-survey for each of the respondents (R1…R14) for each of the dilemmas (D1…D10). The average score for each dilemma is given in the column labeled 'R_avg'; the heuristic's outcome column with 'H'.

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R_avg	H
D1	3	3	3	3	3	3	1	3	2	3	3	3	3	3	**2.8**	**3**
D2	1	1	1	1	1	1	1	1	1	1	3	1	1	1	**1.1**	**1**
D3	3	3	2	3	3	3	2	3	1	3	3	2	3	1	**2.5**	**3**
D4	3	3	3	2	3	3	3	3	2	3	3	1	3	3	**2.7**	**3**
D5	1	1	1	1	1	1	1	3	1	1	3	1	1	1	**1.3**	**1**
D6	1	1	1	1	1	1	3	1	1	1	3	2	1	3	**1.5**	**1**
D7	3	3	3	3	3	3	1	3	1	3	3	2	1	3	**2.5**	**2**
D8	3	1	3	1	3	3	1	1	1	3	3	1	3	1	**2.0**	**3**
D9	3	1	3	2	3	3	1	3	2	3	1	2	1	1	**2.1**	**3**
D10	1	1	1	1	1	2	1	3	1	1	3	1	1	1	**1.4**	**2**

Table 3: Data and Analysis of the Web-Survey (1 = combine, 2 = combine/split, 3 = split)

The correlation between the average respondent score and the heuristic outcome approximately turned out to be 0.810. This is highly a significant result, assuming a two-tailed 99 % confidence interval.

In addition to this first analysis, we examined the relation between the average respondent score and each individual correspondent (**Corr R_avg)**, as well as the relation between the heuristic outcome and each individual respondent (**Corr H**). The respective correlation coefficients are shown in Table 4.

In addition to this analysis, we interviewed respondents R2, R5, R7, R11 and R13 the day following their survey. Of particular interest was the opinion of respondent R11. He explained that in general he is always in favor of splitting up activities in a workflow design in the smallest possible parts. His consideration is that at run-time execution of a workflow process, activities may be dynamically combined to be allocated to workers if this looks like a good idea under the circumstances. Taking a look at the results in Table 3 for respondent R11, this explains the

almost monotonic choice for splitting up activities. Correspondent R7 had different considerations, but could also explicitly support his deviating score.

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
Corr R_avg	0.912 (***)	0.877 (***)	0.855 (***)	0.862 (***)	0.912 (***)	0.875 (***)	0.209	0.482	0.592 (**)	0.912 (***)	-0.048	0.565 (*)	0.706 (**)	0.534 (*)
Corr H	0.890 (***)	0.515 (*)	0.827 (***)	0.579 (**)	0.890 (***)	0.943 (***)	0.000	0.401	0.601 (**)	0.890 (***)	-0.306	0.311	0.749 (***)	0.047

Table 4: Additional Analysis of the Web-Survey (* = sign. 99 %, ** = sign. 95 %, * = sign. 90 %)**

Correspondents R2, R5 and R13 – of whom the individual scores significantly corresponded with our cohesion metric – were much less outspoken about their considerations. They considered the dilemma's one by one, without a general design motive. Correspondent R5 admitted that she was highly intrigued by the relation between her opinion and the cohesion metric, while she could not explicitly support most of her decisions herself in retrospective.

5.2. Discussion

From the second part of the analysis it follows that the opinion of each individual correspondent reasonably well corresponds with the group's average (**Corr R_avg**): The opinion of 11 out of 14 respondents significantly corresponds with this average; for half of the correspondents this significance is high. This gives us some reassurance that comparing the average respondent's score is a good measure for reflecting the group's opinion. Combined with the highly significant correlation between the heuristic and the group's average opinion of 0.810, we cautiously conclude *a positive relation between our cohesion metric and the corresponding intuition of experts on this matter*. Obviously, the set of respondents and questions is very small. Furthermore, we have no hard evidence that the presented dilemmas look like real practical problems; it is based on the author's personal experiences only.

The considerations of the respondents give us some insight into the limits of a cohesion metric like the one we defined. When a design consideration is very specific, the cohesion metric may be a bad implementation. However, when these considerations are less explicit or mixed, then the cohesion metric seems like an attractive and valid quantification thereof.

6. CONCLUSION

On the basis of our evaluation, the author is positive about the value of the cohesion metric for both distinguishing weakly cohesive activities and the support it can offer to decide between design alternatives. Obviously, these results must be interpreted with caution, as discussed earlier. In particular, the interviews with workflow designers showed that very specific design considerations are not well implemented by the cohesion metric. The presented cohesion metric should be used *when more explicit considerations* do not lead to a decisive result. In this sense, it can be used as the 'finishing touch' for a workflow design.

The possibilities to extend this research are many. A following step may be the use of the cohesion metric in question in an actual project, which involves the design of a workflow process in a real setting. Several of the respondents have indicated their willingness to cooperate within such a practical test. It would be a good opportunity to test the heuristic on real design dilemmas.

As stated before, the introduced cohesion metric only *supports* the designer in making a decision with respect to activity definition. It does not suggest any clustering or ordering itself. An extension of the heuristic so that it efficiently generates optimal activity definitions itself is the ultimate but challenging next step of this research. Ideas on the clustering of operations as described by De Sitter (1994) may be an interesting starting point.

Finally, we would like to extend the cohesion metric as described with notions for the 'coupling' degree between several activities. In software engineering, this is another important construct. It gives an indication how modules or classes incorporate a sense of mutual independence. The higher the exchange of calls and information exchange between modules or classes, the lower their independence. Clearly, the notions of coupling and cohesion are related to some level. We suspect that the translation of the concept of coupling to workflow processes may be less straightforward, than it was the case for cohesion. After all, the drawbacks of highly dependent activities seem less severe, than tightly coupled software modules.

Acknowledgments

The authors would like to thank Eric Verbeek of the Technische Universiteit Eindhoven for his assistance in preparing and carrying out the web-based survey.

REFERENCES

W.M.P. van der Aalst. Reengineering Knock-out Processes. *Decision Support Systems*, 30(4):451-468, 2000b.

W.M.P. van der Aalst, H.A. Reijers, and S. Limam. Product-driven Workflow Design. In W. Shen et al., editors, Proceedings of the Sixth International Conference on Computer Supported Cooperative Work in Design 2001, 397-402. NRC Research Press, Ottawa, 2001.

W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, Cambridge, 2002.

S.R. Chidamber, C.F. Kemerer. A Metrics Suite for Object Oriented Design. *IEEE Transactions on Software Engineering*, 20(6): 476-493, 1994.

P.J.N. de Crom and H.A. Reijers. Using Prototyping in a Product-driven Design of Business Processes. In A. D'Atri et al., editors, *Proceedings of the Open Enterprise Solutions: Systems, Experiences, and Organizations Conference*, 41-47. Luiss Edizioni, Rome, 2001.

L. Fischer (ed.). *Workflow Handbook 2001*. Future Strategies, Lighthouse Point, 2001.

P. Grefen, B. Pernici, and G. Sanchez (eds.). *Database Support for Workflow Management: The Wide Project*, Kluwer, Dordrecht, 1999.

P. Grefen, J. Vonk, and P. Apers. Global Transaction Support for Workflow Management Systems: from Formal Specification to Practical Implementation. *VLDB Journal*, 10(4): 316-333, 2001.

T. Harder and A Reuter. Principles of Transaction-Oriented Database Recovery. *Computing Surveys*, 15(4), 287-317, 1983.

S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, London, 1996.

H.A. Reijers. *Design and Control of Workflow Processes: Business Process Management for the Service Industry*. Ph.D. thesis, Eindhoven University of Technology, Eindhoven, 2002. http://tmitwww.tm.tue.nl/staff/hreijers/H.A.%20Reijers%20Bestanden/Proefschrift%20Hajo%20Reijers.pdf

A. Sharp and P. McDermott. *Workflow Modeling: Tools for Process Improvement and Application Development*. Artech House Publishers, Boston, 2001.

A. Seidmann and A. Sundararajan. The Effects of Task and Information Asymmetry on Business Process Redesign. *International Journal of Production Economics*, 50(2-3): 117-128, 1997.

R.W. Selby, V.R. Basili. Analyzing Error-Prone System Structure. *IEEE Transactions on Software Engineering*, 17(2): 141-152, 1991.

L.U. de Sitter. *Synergetic Production.* Van Gorcum, Assen, 1994. (In Dutch)

W. Stevens, G. Myers, and L. Constantine. Structured Design. *IBM Systems Journal*, 13(2): 115-139, 1974.

J. S. Uebersax. A Review of Modeling Approaches for the Analysis of Observer Agreement. *Investigative Radiology*, 27: 738-743, 1992.

J. S. Uebersax. Statistical Methods for Rater Agreement. http://ourworld.compuserve.com/homepages/jsuebersax/agree.htm , 2001.

I. Vessey and R. Weber. Research on Structured Programming: An Emp iricist's Evaluation. *IEEE Transactions on Software Engineering*, 10(4): 394-407, 1984.

E. Weyuker. Evaluating Software Complexity Measures. *IEEE Transactions on Software Engineering*, 14(9): 1357-1365, 1988.