Building a software assessment advisor using expert systems techniques

F. Qiu\(^a\), R.W. Foley\(^b\), R.J. Cole\(^a\)
\(^a\)Software Metrics Laboratory, \(^b\)Department of Computer Studies, Glasgow Polytechnic, Cowcaddens Road, Glasgow G4 0BA, UK

ABSTRACT

An important aspect of the production of quality software is that of software assessment and certification. This involves the measurement of a finished software product using a set of recognised Software Assessment Units. It is essential that at the specification of a piece of software a set of relevant software attributes with a suitable subset of Assessment Units is identified. The set is used to determine whether or not the finished product meets an appropriate quality standard. It has been recognised that a Software Assessment Advisor (SAA) is a necessary tool to assist with this task. An SAA can help in determining the software attributes which should be measured and in selecting an appropriate set of Assessment Units. The SAA will use information concerning the categorisation and functions of a proposed software product, the user's requirements, available tools, and prior assessment experiences in these tasks.

So far, no clear approach has been identified for the development of an SAA. The investigation of the modelling required by an SAA indicates that Expert Systems techniques could provide a useful step forward in this development. Expert Systems techniques are able to relate the information required by the certification procedure intelligently by means of the knowledge model and to support symbolic reasoning to advise the software certifier. Also an Expert System's capability for knowledge learning is significant with respect to the relatively young area of Software Assessment and Certification.

This paper discusses the activities involved in the Assessment and Certification process, describes how an SAA can assist for the process and how an Expert Systems approach has been used in the development of a prototype SAA.

INTRODUCTION

Software quality has become a vital issue in recent times. This is mainly because of the increasing complexity of both software developments and associated technology. There is, therefore, a requirement to ensure that the quality of
Software quality management (SQM) systems are seen as a valuable way to deal with this issue, being defined as: "A program of planned and systematic activities to determine, achieve and maintain required computer software quality"[1]. Quality management activities cover all the activities necessary to assure the quality of software, including such tasks as; the preparation of a software quality management program plan; the development of quality policies, procedures and standards; software quality assurance audits of each phase of development. Another important aspect of quality management is software assessment and certification, where a finished software product is assessed and certified as meeting a measurable standard.

An ESPRIT project which is concerned with the area of software certification is SCOPE [2]. SCOPE aims to define, experiment, and validate European software assessment procedures allowing certification.

There have been two separate approaches used for software assessment leading to certification. These are process certification and product certification. Process certification has been the more traditionally used approach and is really one of indirect certification, relying on the stringency of the software development methodology to assess its quality. One of its main limitations is that this process does not, in any detailed way, address the quantitative characteristics of the software itself. It guarantees that each stage of the development process has been carried out to a specified level but cannot give any real quantitative measure of the actual software which has been produced. The other approach (product certification) seeks to measure directly the functional and non-functional attributes of the developed software and compare them with those of the specification. SCOPE is mainly concerned with product certification. It seeks to take advantage of modern practices in software engineering technology and is to be supported by tools. This enables its automation and therefore enforces its cost effectiveness.

Within the software assessment process, SCOPE has identified an assessment and certification (A&C) method. The stages in this process include:
- submission of the product to a certification agency;
- agreeing on the certification requirement and an initial cost estimate;
- analysing the product specification;
- producing a certification specification;
- relating this specification to tasks, techniques and tools;
- producing, costing and implementing the certification plan;
- producing a certification report.

Because of the complexity of the task, an automated tool, a Software Assessment Advisor (SAA) is highly desirable to provide guidance and assistance to the people who are preparing a plan for assessing a software product. For instance the assessment process involves the selection of a set of relevant software attributes and a suitable subset of assessment units or bricks (a brick enables the application of an assessment technique to software). The selection is based on specific information concerning the software product including the categorisation and functionality of the product, user requirements, available data and tools, and prior assessment experiences. An SAA is a necessary tool to assist with this task.

So far, no clear approach has been identified for the development of an SAA,
although there are some initial studies on it, such as the feasibility study by Wingrove of City University [3], the rule-based framework for A&C description by Hausen et al of GMD [4]. The task itself, however, seems an appropriate one for the application of Expert System techniques and this paper concerns itself with that application. Using such techniques could prove particularly fruitful considering that Software Assessment and Certification is a relatively young area, and the ability of an expert system to refine its knowledge itself to provide better advice with each assessment consultation would be useful.

The following sections discuss the software assessment and certification process identified by SCOPE, how Expert System techniques can be used to assist this process, and the development of an expert system based SAA. A case study example is used to explain the reasoning process of the SAA.

SOFTWARE ASSESSMENT AND CERTIFICATION

The task and approaches

The final objective of certification, as defined by SCOPE, is to independently assess the equivalence between the actual and specified service delivered by software [5]. This is illustrated in Fig.1.

![Assessment task diagram](chart.png)

Fig. 1 Assessment task

Two sets of attributes will be incorporated in the specification. These are:

*Functional Attributes*: which are precisely defined in the specification. Where applicable, existing regulatory constraints will add functional features to the software;

*Non-functional Attributes*: such as reliability, maintainability, security. Which may be specified by the user as specific general quality measures.

As indicated in the section of introduction, two approaches have been used for software assessment leading to certification. Historically, as software was perceived as immaterial, the first effective way to certify it was to rely on the thoroughness of its development methodology. This led to the indirect certification method: *process certification*, with its general principle that the quality of software was in direct relation to the stringency of the development method used, the idea being to give the certification to the software only if an adequate methodology has been used for its development. There are no direct implications for the software, its main limitation being that it does not address the qualitative characteristics of software. It is also difficult to apply properly and is costly.
The other approach, *product certification*, attempts to directly assess the equivalence of functional and non-functional attributes of software at the level of its specifications (specified service), code and behaviour (actual service). As a method of certification, it comprises several items. The key item is the assessment of *equivalence* between specified and actual service. This is based on:

- Examination and measurement of the actual service
- Description and quantification of the specified service.

These involve correctly specifying the different attributes of software. Development and assessment techniques will be dependent on this classification. It includes:

- Operational features, such as type of use (e.g. business, process control, CAD), real time system, importance of graphics, network use, importance of mathematical calculations, use of AI techniques, importance of concurrency;
- Attributes such as correctness, efficiency, integrity, security, safety, reliability, maintainability;
- Software 'white box' characteristics such as size, complexity.

**The SCOPE project**

SCOPE, being mainly concerned with product certification, aims to define, experiment, and validate European software assessment procedures allowing certification. The main expected result of SCOPE is the development of European software assessment technology which comprises sets of methods, techniques and tools.

SCOPE has developed a measurement model which describes how measurement and assessment techniques are organised [6]. The goal is to allow an effective selection and application of these techniques. The current number of the available techniques is extremely large. In order to cope with this complexity, the measurement model is modular: A set of complementary measurement and assessment techniques are encapsulated in an element that is called an 'Assessment Brick'. The basic concept is to consider the brick as a black-box for its selection and the evaluation of its results. Thus the assessment of a software product is performed by applying a series of appropriate assessment bricks. The global pass/fail decision is obtained by evaluating the elementary assessment results from the bricks with regard to the reports.

SCOPE has also identified an Assessment and Certification (A&C) method shown in Fig.2 [7]. Its stages can be summarised as follows:

1) "Submission" involves the presentation of the product specification for A&C;
2) "Requirement" (for assessment) is agreed between supplier and assessor;
3) "Cost estimate" is agreement of the initial assessment cost estimate;
4) "Analysis" produces the definitive profile for the product to be assessed;
5) "Specification" produces the detail of what must be assessed;
6) "Brick selection", enables product parts to be related to assessment units;
7) "Plan" is the step intended to produce the assessment plan;
8) "Cost" is the step during which the cost of A&C is determined;
9) "Assessment" corresponds to the implementation of the A&C plan;
10) "Report" is the final step that produces the report about the assessment.

![Fig. 2 The assessment and certification method](image)

There are a number of case studies in the project, selected for real examples of industrial development of software. The main objective of the case studies is to demonstrate the pragmatic feasibility of the assessment and certification method as a whole, and to assess the predictive success of the method. The bricks mentioned above have been developed through the case studies in phase I of SCOPE project and are being used in the forthcoming case studies of phase II. Some new bricks may also be created during phase II.

Although SCOPE has its achievement of establishing a framework of A&C and has its case studies to demonstrate the feasibility of A&C, there are some gaps between the proposed method and case studies' practices. For example it was proposed that the quality factors required by products, and therefore the attributes to be assessed, should be either indicated by the specification, which is based on the classification mentioned in the section of software assessment and certification, or judged by experts. But the decision of what should be assessed was made in each case study mainly just according to the user requirements and available tools. There is no identified knowledge which could be used as the rules for a classification-based quality factor/attribute selection with experts judgement when necessary. Furthermore it was expected that each brick should be directly or indirectly related to a particular quality factor since the assessment of a software product is performed by applying bricks. But this is not formalised by case studies. Another gap which is of significance to expert systems, is that whilst there is a complete library of all bricks developed in phase I [8], there is very little information, of an easily accessible nature, available to those involved in assessment to assist them in the use of this library in the development of assessment and certification plans.

To fill the gaps a computer aided system would be a useful assistant, based on case studies experiences, to establish logical relations among all related factors or, better, to represent the knowledge involved in the software
assessment process, and to therefore generate a suggestion for an assessment plan. In this respect an Expert System based assistant could provide a solution.

**USING EXPERT SYSTEM TECHNIQUES TO ASSIST SOFTWARE ASSESSMENT**

Although construction of a Software Assessment Advisor (SAA) was not one of the specified tasks within SCOPE, some work has been done within the project. GMD [4] defined a rule-based framework to describe assessment and certification and its application by a rule-system, which might be implemented onto an object-oriented software engineering database system. They expect that an interaction of the particular models is outlined by a predicate specification of an Assessment and Certification Advisor, which might be considered as an abstract method of assessing and certifying software system. A feasibility study of the development of an assessment assistant was made by City University [3]. The authors indicated, "There is no doubt that what we are seeking has all the hallmarks of an expert system. Much of the 'decision making' will be based on 'rules'.". The development of such a system was not the aim of that study. Rather, an Assessment Assistant was proposed which took the form of a hypertext text base organised to guide the assessor, step by step, through the stages of the Assessment and Certification Method leading to the production of the Assessment Plan. However, the investigation clearly highlighted the suitability of the application area to Expert Systems (ES).

Because of the properties of Expert Systems, the use of ES techniques could provide better assistance for software assessment and certification. An Expert system is a computer program which simulates the method and the steps through which human experts use their domain-specific knowledge to solve problems. There are several key components of ES including a knowledge base, current database, inference engine, explanation and knowledge learning mechanisms. The basic architecture of an expert system is depicted in Fig.3.

![Fig. 3 Basic architecture of an expert system](image)
The Knowledge base is for the storage of domain knowledge. The knowledge representation takes a certain form such as rule-based, frame-based, object-oriented form. The specific representation depends on the way in which the large body of domain-specific knowledge can conveniently be stored in data structures for the purposes of symbolic computation. Symbolic computation means non-numeric computations in which the symbols can be construed as standing for various concepts and relationships between them. The Current database is used to store the initial data and intermediate information for the current problem. The Inference engine is composed of a set of procedures for scheduling and controlling the whole system and providing the solution to the user. The solution's deduction is based on the data in the current database, the knowledge in the knowledge base and assigned reasoning strategies. The explanation function is performed by a set of procedures to explain the actions of the system to the user. In general it answers questions about why some conclusion was reached or why some alternative was rejected. A Knowledge learning mechanism is used for updating and enhancing the knowledge in the knowledge base with each consultation or as further domain knowledge becomes available.

An expert system based SAA should have advantages in assisting the software assessment and certification process. It is able to provide a suitable form to represent the knowledge involved in the process. For example, the knowledge

"when the software product is required for a long life cycle, the attribute 'maintainability' will be assessed"

could be represented by a rule of the form,

if there is a long life cycle requirement then 'maintainability' should be assessed.

The knowledge of a software product could be represented as a frame. A frame is a structure consisting of a network of nodes (or slots), with each slot containing a property about the frame entity. The software product frame would include slots for categorisation, function, programming characteristic, size etc. to describe each aspect of the product. Then, according to the system's reasoning strategy, relevant pieces of knowledge will be selectively applied for producing the advice. It can also provide explanations for conclusions reached. Finally its capacity to update the knowledge to provide better advice is particularly significant with respect to the area of Software Assessment and Certification, where actual A&C expert knowledge is itself limited and is continually being refined.

DEVELOPMENT OF AN EXPERT SYSTEM BASED SOFTWARE ASSESSMENT ADVISOR (SAA)

Knowledge model used for SAA prototype
We require an expert system-based SAA to guide the selection from a set of relevant software attributes and a suitable subset of assessment units or bricks. The knowledge model developed for implementing this SAA, as shown in Fig.4, is composed of the following components:
"Product description" relates to step 1 of the A&C method shown in Fig.2. The description includes product categorisation, purpose, functions, characteristics and effort.

"User requirement" relates to steps 2-4, and considers all the users' requirements during the process of making the selections.

"Prior assessment experience" contains knowledge about selected case study experiences, currently containing 6 case studies of SCOPE [9]. This portion of the knowledge model will benefit from the knowledge learning ability of the expert system as further case study data becomes available.

"Attribute selection" relates to step 5, and determines the selection of the software attributes to be assessed (such as reliability, usability,...). This selection involves the knowledge of product description, user requirement, tools confidence and prior assessment experience.

"Brick selection" relates to step 6, and involves the construction of the assessment plan, based on the knowledge of selected attributes, available data, tools support, user requirement and prior experience.

Based on this knowledge model, the prototype's system major functional components are constructed as shown in Fig.5.
Knowledge Representation
To best represent the domain knowledge, a Hybrid Knowledge Representation is adopted. There are essentially two types of domain knowledge: the procedural knowledge which includes the attribute and brick selection; and the declarative knowledge which includes the knowledge of product categorisation and the prior assessment experiences from the case studies. For the procedural knowledge a rule-based form is used, whilst for the declarative knowledge a frame-based form is used.

Procedural Knowledge Representation  The rule-based representation of the procedural knowledge uses a series of weights which affect the inference process itself. For example, the knowledge of attribute selection is represented as a rule that captures the relative importance of features that include: product function, categorisation and characteristic; the user; associated tools; and prior assessment record. These four features are ranked in relative importance by respective weightings 0.5, 0.3, 0.1, 0.1, to add to one. In this example, the following rule is used to determine whether or not a particular attribute $A_k$ should be assessed,

$$\text{If } (W_{Ak} \times \sum_{j} W_{Skj}) \geq 0.5, \text{ then the attribute } A_k \text{ should be assessed.}$$

where $W_{Ak}$ (the weight of the attribute $A_k$) is a function calculated by the system from each importance weight $W_{Akj}$ (details given in appendix) and $W_{Skj}$ are selection weights, each restricted to the range $[0, 1]$ (the weight $WF_{ij}$, used later, is also an importance weight). The value 0.5 in the antecedent of the rule is a weighting threshold, above which it is considered that the attribute $A_k$ is sufficiently relevant to a particular case that it should be selected for measurement. The attribute selection knowledge is broken down further by related rules of the type:

If the assessment of attribute $A_k$ is required by product function $F_i$ when the product categorisation is $C_j$, and/or the assessment of $A_k$ is required by product characteristic $CH_q$,

then $A_k$ has a selection weight

$$W_{Sk1} = \begin{cases} 0.5 \times \min \{WF_{ij}, 1\}, & \text{when } A_k \text{ only required by } F_i, i \in \{1, 2, \ldots\} \\ 0.5 & \text{others} \end{cases}$$

If the assessment of attribute $A_k$ is required by the user,

then $A_k$ has a selection weight

$$W_{Sk2} = 0.3;$$

Other rules, including those dealing with brick selection, take a similar form.
Declarative Knowledge Representation

For the declarative knowledge, i.e. the knowledge of product categorisation and prior assessment experience, the representation takes a frame-based form. For example, the main knowledge from case study X-RAYS [10], which is a process monitoring system, is represented as shown in Fig.6.

![Frame-based representation of X-RAYS](image)

(* references gct..., given under bricks, are references to specific assessment units [11])

In the reasoning of the system, if the 'Product description' portion of the frame to be instantiated sufficiently matches (through the use of well-established frame based matching strategies [12]) with the product to be assessed, then the 'Assessment experience' portion is to be used as an assessment reference.

Reasoning Procedures

The reasoning procedure is illustrated as shown in Fig.7. It consists of 4 subprocedures:

- attribute selection,
- brick selection,
- identifying the closest case study experience, and
- knowledge updating.

The task of the reasoning scheduler module is to start and schedule each reasoning subprocedure. When product assessment advice is required, it first starts rule-based deduction for the attribute selection. If the selection is successful, and a series of attribute weights reach the threshold value, then the subprocedure for brick selection will be started to allow the system to find an
appropriate set of bricks to measure the selected attributes. After these two procedures a suitable solution should have been deduced to advise the assessor. If, however, the result in any of these two procedures is not satisfactory, or if the user specifically requests supplementary information, the third procedure, which attempts to identify a case study product closest to the product to be assessed, is invoked. This procedure uses a reasoning strategy of matching value calculation [12] to identify a prior assessed product. Then the assessment experience based on the identified case study will be given as advice.

![Diagram](image)

**Fig. 7 Reasoning procedure**

**Knowledge Learning**
Knowledge learning, which is a useful technique in expert systems, allows the system to refine its knowledge based on the accumulation of case study experiences. As we have seen, the knowledge base maintains a series of importance weights which are used in the rule-based deduction. These weights are calculated statistically from the existing case study information. As each new product completes its assessment, or as additional case study material becomes available, the weights are updated and revised. Refinement of these weights allows the system to enhance its knowledge to provide better advice with each assessment consultation.
802 Software Quality Management

The SAA maintains 3 specific categories of importance weights, these are; product function weights (WF$_{ij}$); software attribute weights (WA$_{ki}$); and assessment brick weights (WB$_{qk}$). The definitions ensure that each weight range is in $[0, 1]$.

Suppose that, for the prior assessment experience knowledge,

NC$_j$ is the number of occurrences (noc) of product categorisation C$_j$,
NF$_{ij}$ is the noc of product function F$_i$ when the product categorisation is C$_j$,
NA$_{ki}$ is the noc of attribute A$_k$ when attribute A$_k$ was selected and related to F$_i$,
NB$_{qk}$ is the noc of brick B$_q$ when brick B$_q$ was selected and the selection related to A$_k$.

Then,

\[
WF_{ij} = \frac{NF_{ij}}{NC_j} \quad (i=1,2,...; j=1,2,...);
\]

\[
WA_{ki} = \frac{NA_{ki}}{NF_i} \quad (k=1,2,...), \quad \text{where} \quad NF_i = \sum_{j} NF_{ij};
\]

\[
WB_{qk} = \frac{NB_{qk}}{NA_k} \quad (q=1,2,...), \quad \text{where} \quad NA_k = \sum_{i} NA_{ki}.
\]

When the SAA is in operation, the search strategy of the reasoning procedure first considers those factors whose related importance weights are greater than or equal to the predetermined threshold in its decision making process. This threshold is determined on the basis of the prior assessment experiences. WF$_{ij}$ and WA$_{ki}$ will also affect the decision of attribute selection, WB$_{qk}$ will affect the brick selection.

AN EXAMPLE OF THE PROCESS USING A CASE STUDY

Let us use a case study example to more fully illustrate the above process.

Suppose that software product X-RAYS, mentioned in the subsection of knowledge representation, is the product to be assessed.

Through the user interface, the SAA would obtain relevant information as following:

- the description of the product (product description part in Fig.6),
- the assessment of attribute 'correctness'(A$_1$) is required by the user.

Then by reference to the Characteristic-Attribute and Function-Attribute sections of the knowledge base, the SAA finds that,

- the product characteristic 'a bespoke project'(CH$_1$), has requirement relations with software attributes: maintainability (A$_2$) and testability (A$_3$),

- the product programming characteristic 'algorithm model'(CH$_2$) has a
requirement relation with attribute A1,

- the product function 'action generation'(F1) has requirement relations with A1 and attribute 'useability'(A4) respectively,

- the product function 'data communication'(F2) requires attribute A4,

- the product function 'status signals synthesis'(F3) requires attribute 'efficiency'(A5) and 'reliability'(A6).

Now SAA uses A1, A2, A3, A4, A5, A6 as indices to search the records in the Tool support, Prior assessment experience and Weight sections of the knowledge base sequentially. The results obtained are:

A1  - no tools identified, some prior assessment records,
     WF11=0.7, CH21=0.85 WA11=0.78 ;

A2  - there are some tools, no prior assessment record,
     CH12=0.9, WA2=0.9 ;

A3  - there are some tools, some prior assessment records,
     CH13=0.8, WA3=0.8 ;

A4  - no tools identified, some prior assessment records,
     WF11=0.7, WF21=0.6, WA41=0.67, WA42=0.71 ;

A5  - no tools identified, some prior assessment records,
     WF31=0.8, WA53=0.8 ;

A6  - there are some tools, some prior assessment records,
     WF31=0.8, WA63=0.9 .

Considering attribute A1 first, its corresponding weights are

\[ WS_{11}=0.5 \times 0.7 = 0.35 \]
\[ WS_{12}=0.3 \]
\[ WS_{13}=WS_{14}=0 \]
\[ WA_1=CH_{21}=0.85 \]

and therefore its threshold antecedent is given as,

\[ WA_1 \times \sum WS_{1j} = 0.85 \times 0.65 = 0.5525 \]

Similarly for all the other attributes

\[ WA_2=0.9, \sum WS_{2j}=0.6, \quad WA_2 \times \sum WS_{2j}=0.54 ; \]
\[ WA_3=0.8, \sum WS_{3j}=0.7, \quad WA_3 \times \sum WS_{3j}=0.56 ; \]
\[ WA_4=0.71, \sum WS_{4j}=0.5, \quad WA_4 \times \sum WS_{4j}=0.36 ; \]
\[ WA_5=0.8, \sum WS_{5j}=0.5, \quad WA_5 \times \sum WS_{5j}=0.4 ; \]
\[ WA_6=0.9, \sum WS_{6j}=0.6, \quad WA_6 \times \sum WS_{6j}=0.54 . \]

Therefore the SAA suggests that software attributes A1, A2, A3, A6 should be
assessed as their weights are greater than the 0.5 threshold.

Next, the system scheduler uses the results as the input to start the deduction for brick selection. Using its similar reasoning method, the following Bricks would be suggested:

- grs1v0.0.1- operational correctness
- grs3v0.0.1- functional correctness
- gct1v0.0.2- testability
- gct4v0.0.1- maintainability
- etn3v0.0.1- consistency

The reasoning procedure for this product assessment is basically complete. However, if a prior assessment record was required, for reference, by the user, the scheduler would start the third reasoning procedure and a closest example would be made available.

Finally, the relevant information about this assessment is used for knowledge updating and added to the knowledge base. For instance the importance weights WF_{11} and WA_{11} are updated as following:

Before updating, these weights were as follows:

\[ WF_{11} = 0.7, \quad WA_{11} = 0.78; \]

where NC_{1}=10, NF_{11}=7, NA_{11}=7, NF_{1}=9 .

After updating:

\[ WF_{11} = \frac{7+1}{10+1} = 0.727, \]
\[ WA_{11} = \frac{7+1}{9+1} = 0.8; \]

and NC_{1}=11, NF_{11}=8, NA_{11}=8, NF_{1}=10 .

WA_{41} was also identified by F_{1} but not selected. Thus that value of WA_{41} is reduced from 0.67(NA_{41}=6) to (6+0)/(9+1)=0.6 .

All the other weights which are relative to the assessment are changed in a similar way.

SUMMARY

This paper introduces the task of software assessment and certification, and the method identified by the ESPRIT project SCOPE. It discusses the approaches to computer aided systems as assessment assistants and the advantages of using ES techniques in this area. After outlining the basic properties of Expert systems, a prototype system, which is currently under development, is described. This system using a hybrid knowledge structure with relative reasoning method and an automatic knowledge updating process. An expert system based software assessment advisor discussed here could assist with the first stage in the assessment and certification process to give advice about what attributes should be assessed and what bricks and tools could be used. As further case study
results become available, the system will be evaluated, adapted, and enhanced. Future system extensions could consider more factors related to selection including assessment and quality standard levels.

REFERENCES


APPENDIX

Details of the calculation of the function WAk:

\[
WA_k = \begin{cases} 
\max\{\max\{WA_{ki}\}, \max\{CH_{qk}\}\}, & \text{when } A_k \text{ is identified only by } F_{ij}\text{ or/and } CH_q, i,j,q\in\{1,2,\ldots\}; \\
1/n \sum_{i=1}^{n} WA_{ki}, & \text{when } A_k \text{ is identified only by user}; \\
\max\{\text{above}\}, & \text{others}.
\end{cases}
\]

where \(CH_{qk}\) is a relation weight of \(CH_q\) and \(A_k\)