Towards a Theory of Web Effort Estimation

Emilia Mendes[1], Simon Baker[2]

[1] Computer Science Department, The University of Auckland,
Private Bag 92019, Auckland, New Zealand
emilia@cs.auckland.ac.nz
Bolsista CAPES/Brazil at COPPE/Sistemas, Federal University of Rio de Janeiro, Brazil
[2] Global Business Services, IBM
55 Pyrmont Street, Sydney, 2009, NSW, Australia
bakesim@au1.ibm.com

Abstract. Reliable Web effort estimation is one of the cornerstones of good Web project management. Hence the need to fully understand which factors affect, and how they affect, a project's outcome, and their causal relationships. The aim of this paper is to propose a mechanism to obtain an empirical generalization of four different causal process form type of theories for Web effort estimation. Each one of these theories was constructed using a hybrid approach to theory building, based on existing knowledge elicited from several domain experts, data on past completed Web projects, and a technique that enables the modeling of causal relationships and their uncertainty. The aggregation methodology used to combine these theories was the same one we employed previously with causal maps, which extends previous work by adding a mapping scheme to handle complex domains (e.g. effort estimation), and in addition also uses an aggregation process that preserves all the causal relations from the original theories. The resultant empirical generalization contains 70 concepts (variables), and highlights several patterns amongst the four theories used as input, thus providing a starting point to a more general theory of Web effort estimation.

Keywords: Web effort estimation, causal maps, Web effort prediction, map aggregation.

1 Introduction

The topic relating to the research detailed herein – Web effort estimation, falls within the field of "Web Engineering"[1], term that, despite being first published in 1996 in a conference paper by Gellersen et al [19], was only defined in 2001 by Murugesan and Deshpande [51], as follows:

> *"the use of scientific, engineering, and management principles and systematic approaches with the aim of successfully developing, deploying and maintaining high quality Web-based systems and applications".*

Scientific knowledge represents a "system for description [of concepts] and explanation [of why events occur]" [55], which should provide: i) a typology, or system of classification - method that organises and categorises concepts (variables); ii) predictions of future events; iii) explanations of past events; iv) a sense of understanding about what causes these events;

[1] We would also like to point out that in our view Web and software development differ in a number of areas, such as: Application Characteristics, Primary Technologies Used, Approach to Quality Delivered, Development Process Drivers, Availability of the Application, Customers (Stakeholders), Update Rate (Maintenance Cycles), People Involved in Development, Architecture and Network, Disciplines Involved, Legal, Social, and Ethical Issues, and Information Structuring and Design. A detailed discussion on this issue is provided in [31].

and optionally v) the potential for control of phenomena. Within the scope of this work we also support the view put forward by Dubin that the terms *model* and *system* can be used as synonyms of *theory* [10].
A scientific body of knowledge is generally built using one of two strategies: Research-Then-Theory, and Theory-Then-Research [55].
The Research-Then-Theory strategy consists of the following steps [55]:
1. To select a phenomenon and list all of its characteristics;
2. To measure all the characteristics of the phenomenon identified in step 1) in as many as possible varied situations;
3. To analyse very carefully the data gathered in step 2) in order to verify whether there are any regular patterns in the data that deserve further attention;
4. To formalise the patterns identified in step 3) as theoretical statements (e.g. laws, axioms, propositions, hypotheses and empirical generalisations) [55].

And the Theory-Then-Research strategy comprises the following sequence of tasks:
1. To build up an explicit theory in either axiomatic or process description form;
2. To choose a statement produced by the theory created in step 1) to be compared with the results from empirical research (validating the statement);
3. To carry out an empirical investigation in order to assess the correspondence between the statement chosen in step 2) with the results from empirical research;
4. To return to step 2) in case there are other statements to be chosen for validation OR return to step 2) in case the research results from step 3) do not support the statement investigated in that step. Prior to returning to step 2) changes need to be made to either the theory, or to the empirical investigation's design. It is also applicable during this step to identify the situations to which the theory developed in step 1) does not apply.

Although these two strategies are widely used, both present problems. Here we will only document the problems considered to be the most fundamental for each of these strategies; however a detailed discussion is provided in [55]:
Research-Then-Theory: The great difficulty in identifying up front all the variables that should be measured within the context of the phenomenon being investigated, given this can be an awfully long list; and the enormous challenge in selecting the significant causal relationships amongst all possible relationships.
Theory-Then-Research: The great difficulty in inventing the initial theory.

Reynolds [55] suggests the use of a composite (hybrid) approach to scientific theory building, where scientific activity is split into three stages – Exploratory, Descriptive, and Explanatory. Each is detailed below:
1. *Exploratory*. This step entails one to freely explore the phenomenon under investigation, and to develop suggestive ideas. This activity differs from that employed in the usual Research-Then-Theory strategy because here the data that is gathered is very much influenced by the investigator's hunches and insights. Ideally this step should also provide direction for the actions to take place during stage 2.
2. *Descriptive*. This step entails building up explicit descriptions of patterns that were conjectured during step 1). The purpose may be seen as one of developing inter-subjective descriptions, i.e., descriptions of a phenomenon that were personally

experienced (subjectively) by more than one subject. These are also understood as empirical generalisations. These descriptions are then used as input to the next step, where a theory is built up.

3. *Explanatory*. This step entails the refinement of a theory that can be used to explain the descriptions identified in step 2). This is a continuous cycle of:
    a. Theory construction;
    b. Theory testing, where empirical research is used in an attempt to falsify the proposed Theory;
    c. Theory reformulation, back to step 3a.

The main research contribution of this paper is to propose a mechanism to obtain an empirical generalisation of four different causal process form type of theories for Web effort estimation. However, each of these four different theories are also research contributions, as no previous study to date in either Web or Software Engineering has argued that a validated Bayesian Network model can be taken as a causal process type of theory.

Web effort estimation is the process by which effort is forecasted and used as basis to predict project costs and allocate resources effectively, so enabling projects to be delivered on time and within budget [1]. It is one of the cornerstones of Web project management, and a very complex domain where the relationship between factors is non-deterministic and has an inherently uncertain nature.

We employed Reynolds' composite approach to construct each of those theories, based on existing knowledge elicited from separate single-company Web effort estimation domain experts, data on past completed Web projects, and a technique that enables the modeling of causal relationships and their uncertainty. The empirical generalisation presented herein summarises, in general form, the patterns in each of the theories used as input to this generalisation. In order to use the aggregation mechanism we employ to 'transform' the resulting empirical generalisation into a causal process form type of general theory of Web effort estimation we would also need to: i) 'merge' the probabilities from all input theories, and ii) validate the theory empirically using hard evidence from past projects. In relation to item ii), we have all the validation sets employed when validating each of the four theories used herein; however, when it comes to item i), this is unfortunately a task that at this stage is impossible to carry out given the diverse quantifications from the original theories.

We also conjecture that as more theories are aggregated to the initial empirical generalisation, the greater the confidence one has that the patterns that were observed will be repeated in a concrete situation in the future if the same conditions were to recur.

The remainder of this paper is structured as follows: Section 2 provides readers with some background on Web effort estimation and related work, followed by an introduction to Bayesian Networks and the methodology employed to build each of the four causal process form type of theories in Section 3. Section 4 describes the aggregation mechanism employed to create the empirical generalisation, followed by the results from this empirical generalisation in Section 5, comments on threats to the validity of our work in Section 6, and finally conclusions in Section 7.

2 Background & Related Work

Web development is a rapidly growing industry, where the number of Web development companies in the United States alone increased from less than 1000 businesses in 1995 to over 30,000 in 2005 [22]. In addition, by 2010 the Web development industry was expected to experience a further growth of over 20%. This suggests that some of the existing research in Web Engineering should be geared towards helping Web companies understand, control and improve their current processes. One of such processes is Web effort estimation.

There have been numerous previous attempts to model effort estimation of Web projects, but none to date aimed at building a theory of Web effort estimation. Mendes and Counsell [32] were the first to investigate this field by building a model using machine-learning techniques with data from student-based Web projects, and size measures harvested late in the project's life cycle. Mendes and collaborators also carried out a series of consecutive studies (e.g. [7]-[16],[25]-[49]) where models were built using multivariate regression and machine-learning techniques and used data on industrial Web projects. Later they proposed and validated size measures harvested early in a project's life cycle, therefore better suited to effort estimation [43] when compared to other Web effort predictors previously proposed [25]. Reifer [54] proposed an extension of an existing software engineering resource model, and a single size measure harvested late in the project's life cycle. None were validated empirically. This size measure was later used by Ruhe et al. [56], who further extended a software engineering hybrid estimation technique to Web projects, using a small data set of industrial projects, with this technique mixing expert judgement and multivariate regression. Baresi et al. [4],[5] and Mangia et al. [24] investigated effort estimation models and size measures for Web projects based on a specific Web development method. More recently there have been a number of studies describing causal maps for Web effort estimation [2],[26]-[29], where, except for [2], their causal relationships were identified by a domain expert, using only the set of factors that are part of the Tukutuku database [38]. Baker and Mendes [2] proposed a mechanism to aggregate several causal maps for Web effort estimation to help Web companies elicit their tacit knowledge relating to effort estimation. This same mechanism is employed herein to obtain an empirical generalisation from four Web effort estimation theories. Other more recent studies compared Web effort prediction techniques, based on existing datasets [7]-[13].

None of these previous studies investigated explicitly the issue relating to building a theory of Web effort estimation. In addition, most studies, when identifying important factors for Web effort estimation, focused solely on factors that presented a cause & effect relationship with effort, i.e., they included any factors correlated with effort. In addition, when surveying companies to identify suitable effort predictors, those studies did not assess how good these companies were at estimating effort for their Web projects.

As part of a NZ government-funded research, Mendes elicited several company-specific expert-driven Web effort estimation causal maps from NZ Web companies [26]. The elicitation process employed is detailed in [26], and also introduced in Section 3. A total of 10 causal maps were elicited with Web companies in New Zealand; many were later used as part of corresponding larger models, built using Bayesian Networks (technique detailed later), each providing a representation of the Web effort estimation domain from the perspective of the single Web company from which that model had been elicited. All participating companies were consulting companies that developed different types of Web applications (e.g. static application, applications that used a content management system, database-driven Web applications). Four Bayesian models were built and validated empirically. These are the ones used herein. Within the context of this work we conjectured that consulting companies would have experience managing a much broader range of Web projects, thus contributing more strongly towards more 'generic' representations of the Web effort estimation phenomenon, when compared with Web companies that do not provide consulting services.

3 Building Bayesian Networks

3.1 Introduction to Bayesian Networks

A Bayesian Network (BN) is a probabilistic model that allows for reasoning under uncertainty. A BN is made up of two components [21]. The first is a graphical causal map, depicted by a Directed Acyclic Graph (DAG) (see Figure 1). The DAG's nodes represent the relevant variables (factors/concepts) in the domain being modelled, which can be of different types (e.g. observable or latent, categorical). The DAG's arcs represent the causal relationships between variables, where relationships are quantified probabilistically. These graphs may be simple (as in the example in Figure 1), or very complex in terms of nodes and relations.

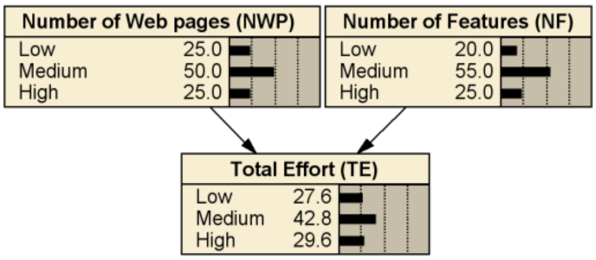


Fig. 1. A small Bayesian Network model and its CPTs

The second component of a BN is the quantitative part: a Conditional Probability Table (CPT) associated to each node in the network. A parent node's CPT describes the relative probability of each state (value) (Fig. 1 CPTs for nodes 'Number of Web pages' and 'Number of Features'); a child node's CPT describes the relative probability of each state conditional on every combination of states of its parents (Fig. 1 CPT for node 'Total Effort'). Each row in a CPT represents a conditional probability distribution and therefore its values sum up to one [21].

Once a BN is specified, evidence (e.g. values) can be entered into any node, and probabilities for the remaining nodes automatically calculated using Bayes' rule [21]. Therefore BNs can be used for different types of reasoning, such as predictive and "what-if" analyses to investigate the impact that changes on some nodes have on others [13]. A description of the generic process employed to build the four BNs part of the research presented herein, and its correspondence to Reynolds' composite approach is detailed below.

3.2 Process Employed to Build Bayesian Networks

The process employed to build all four BNs is an adaptation of the Knowledge Engineering of Bayesian Networks (KEBN) process proposed in [60] (see Fig. 2). In Fig. 2 arrows represent flows through the different processes, depicted by rectangles. Such processes are executed either by people – the Knowledge Engineer (KE) and the Domain Experts (DEs) (white rectangles), or by automatic algorithms (dark grey rectangles). Within the context of this paper the first author was the KE, and Web project managers from four well-established Web companies in Auckland were the DEs. This Figure also presents the correspondence between the phases that are part of the adapted KEBN process, and Reynolds' stages. Note that the adapted KEBN process enables iteration through all phases, whereas Reynolds' approach only prescribes iteration during the Explanatory stage, and a sequential flow from the Exploratory to the Descriptive stage, and from the Descriptive to the Explanatory stage. However, we contend that when applied to a real situation, the sequential flow is replaced by an iterative flow as part of theory construction. This was indeed what we have observed in practice.

The three main steps within the adapted KEBN process are the Structural Development, Parameter Estimation, and Model Validation (see Fig. 2). This process iterates over these steps until a complete BN is built and validated. Each of these three steps is detailed below:

Structural Development: This step represents the qualitative component of a BN, which results in a graphical structure comprised of, in our case, the factors (concepts, variables) and causal relationships identified as fundamental for effort estimation of Web projects. In addition to identifying variables, their types (e.g. query variable, evidence variable) and causal relationships, this step also comprises the identification of the states (values) that each variable should take, and if they are discrete or continuous. In practice, currently available BN tools require that continuous variables be discretised by converting them into multinomial variables [60], also the case with the BN software used in this study. The BN's structure is refined through an iterative process. This structure construction process has been validated in previous studies [12][13][23][60] and uses the principles of problem solving employed in data modelling and software development [58]. During this step, both the brainstorming that takes place and also the decisions that are made (e.g. choice of variables and discretisation) are documented using a digital voice recorder and a text editor. This is done in order to provide evidence relating to the explicit descriptions of the models (patterns) identified during this step. The evidence obtained corresponds to the actions that characterise the Descriptive step, according to Reynolds' approach.

As will be detailed later, existing literature in Web effort estimation, and knowledge from the domain experts were employed to elicit the Web effort BNs' structure. Throughout this step the knowledge engineer(s) also evaluate(s) the structure of the BN, done in two stages. The first entails checking whether [60]: variables and their values have a clear meaning; all relevant variables have been included; variables are named conveniently; all states are appropriate (exhaustive and exclusive); a check for any states that can be combined. The second stage entails reviewing the BN's graph structure (causal structure) to ensure that any identified d-separation dependencies comply with the types of variables used and causality assumptions. D-separation dependencies are used to identify variables influenced by evidence coming from other variables in the BN [21]. Once the BN structure is assumed to be close to final knowledge engineers may still need to optimise this structure to reduce the number of probabilities that need to be elicited or learnt for the network. If optimisation is needed then techniques that change the causal structure are employed [12][21].

Parameter Estimation: This step represents the quantitative component of a BN, where conditional probabilities corresponding to the quantification of the relationships between variables are obtained. Such probabilities can be attained via Expert Elicitation, automatically from data, from existing literature, or using a combination of these. When probabilities are elicited from scratch, or even if they only need to be revisited, this step can be very time consuming. In order to minimise the number of probabilities to be elicited some techniques have been proposed in the literature [11][12][59].

Model Validation: This step validates the BN that results from the two previous steps, and determines whether it is necessary to re-visit any of those steps. Two different validation methods are generally used - Model Walkthrough and Predictive Accuracy. This step corresponds quite clearly to the Explanatory step suggested by Reynolds, given that the data employed to validate the BN model (proposed Theory) comprises the empirical research used in an attempt to falsify this theory; in addition, the data is also used to re-calibrate the initial model (whenever needed), and this can be taken as representing a theory reformulation.

Model walkthrough represents the use of real case scenarios that are prepared and used by domain experts to assess if the predictions provided by a BN correspond to the predictions experts would have chosen based on their own expertise. Success is measured as the frequency with which the BN's predicted value for a target variable (e.g. quality, effort) that has the highest probability corresponds to the experts' own assessment.

Predictive Accuracy uses past data (e.g. past project data), rather than scenarios, to obtain predictions. Data (evidence) is entered on the BN model, and success is measured as the frequency with which the BN's predicted value for a target variable (e.g. quality, effort) that has the highest probability corresponds to the actual past data. However, previous literature also documents a different measure of success, proposed by Pendharkar et al. [52], and later used by Mendes [27][26][29], and Mendes and Mosley [28]. Herein, an effort point forecast for each past project being used for validation is obtained by computing estimated effort as the sum of the probability (ρ) of a given effort scale point multiplied by its related mean effort (μ), after normalising the probabilities such that their sum equals one. Therefore, assuming that Estimated Effort is measured using a 5-point scale (Very Low to Very High), we have:

$$Estimated\,(Effort) = \rho_{VeryLow}\mu_{VeryLow} + \rho_{Low}\mu_{Low} + \rho_{Medium}\mu_{Medium} + \rho_{High}\mu_{High} + \rho_{VeryHigh}\mu_{VeryHigh} \quad (1)$$

Within the context of Web effort estimation and to some extent software effort estimation, the challenge using Predictive Accuracy is the lack of reliable effort data gathered by Web and Software companies. Most companies, who claim to collect effort data, use manually entered electronic timesheets (or even paper!) which is unreliable when staff rely on their memory and complete their timesheets at the end of the day. Collecting manually entered timesheets every 5 minutes (assume 1 minute/entry) in a bid to improve data accuracy increases data collection cost by as much as 10 fold. The problem here is that "effort accuracy" is inversely related to productivity, i.e., the longer one takes filling out timesheets the less time one has to do the real work!

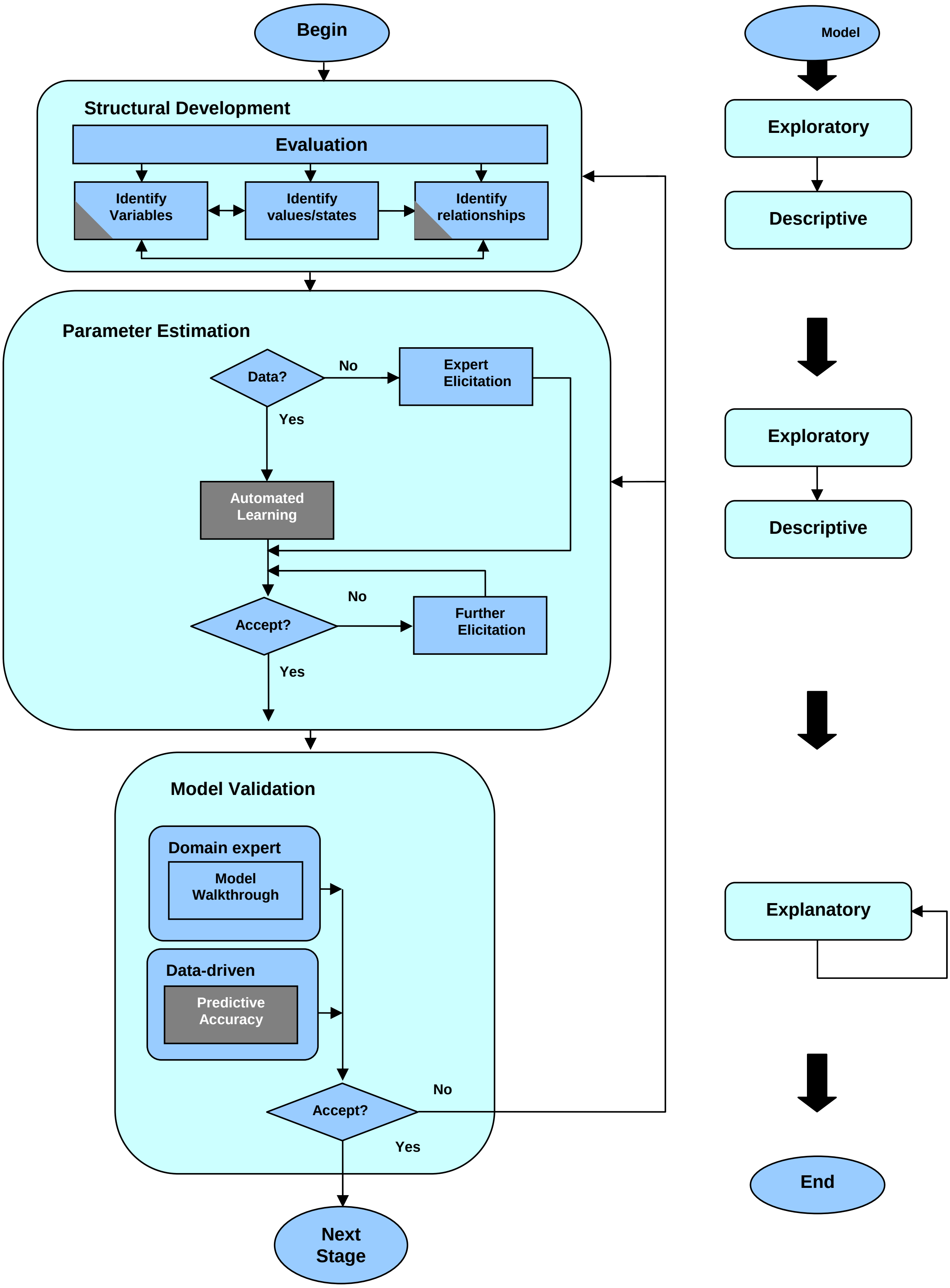


Fig. 2. Adapted KEBN process and corresponding stages of Reynolds' composite approach

3.3 Process Employed to Build Bayesian Networks

This sub-Section revisits the adapted KEBN process (see Fig. 2), detailing the tasks carried out for each of the three main steps that form part of that process. Before starting the elicitation of the Web effort BN model, the Domain Experts (DEs) from all participating Web companies were presented with an overview of Bayesian Network models, and examples of “what-if” scenarios using a made-up BN. This, we believe, facilitated the entire process as the use of an example, and the brief explanation of each of the steps in the KEBN process, provided a concrete understanding of what to expect. We also made it clear that the knowledge Engineers were facilitators of the process, and that the Web companies’ commitment was paramount for the success of the process. The effort required by each company to have their BN models created, and the characteristics of each model, are detailed in Table 2.

Table 1. Characteristics of the BNs created & companies involved

	Company B	Company C	Company G	Company H
Number of DEs	1	1	2	2
Number of 3-hours elicitation sessions	12	6	8	12
Total hours to elicit & validate model	36 hours	18 hours	24 hours	36 hours
Effort to elicit & validate model	72 person/hours	36 person/hours	72 person/hours	108 person/hours
Number of factors	14	13	34	33
Number of relationships	18	12	41	60
Validation set	22 projects	8 projects	11 projects	22 projects

The DEs who took part in this case study were project managers of four well-established Web companies in Auckland (New Zealand). All four companies were of small size, where all of their project managers had each worked in Web development for at least 10 years. In addition, all four companies developed a wide range of Web applications, from static & multimedia-like to very large e-commerce solutions. They also used a wide range of Web technologies, thus enabling the development of Web 2.0 applications. They were all looking at improving their current effort estimates.

Detailed Structural Development and Parameter Estimation: In order to identify the fundamental factors that the DEs took into account when preparing a project quote we used the set of variables from the Tukutuku dataset [43] as a starting point (see Table 1). We first sketched them out on a white board, each one inside an oval shape, and then explained what each one meant within the context of the Tukutuku project. Our previous experience eliciting BNs in other domains (e.g. ecology, resource estimation) suggested that it was best to start with a few factors (even if they were not to be reused by the DE), rather than to use a “blank canvas” as a starting point [29].

Table 2. Tukutuku variables

	Variable Name	Description
Pr oje	*TypeProj*	Type of project (new or enhancement).
	nLang	Number of different development languages used

	DocProc	If project followed defined and documented process.
	ProImpr	If project team involved in a process improvement programme.
	Metrics	If project team part of a software metrics programme.
	DevTeam	Size of a project's development team.
	TeamExp	Average team experience with the development language(s) employed.
Web application	*TotWP*	Total number of Web pages (new and reused).
	NewWP	Total number of new Web pages.
	TotImg	Total number of images (new and reused).
	NewImg	Total number of new images created.
	Num_Fots	Number of features reused without any adaptation.
	HFotsA	Number of reused high-effort features/functions adapted.
	Hnew	Number of new high-effort features/functions.
	TotHigh	Total number of high-effort features/functions
	Num_FotsA	Number of reused low-effort features adapted.
	New	Number of new low-effort features/functions.
	TotNHigh	Total number of low-effort features/functions

Within the context of the Tukutuku project, based on collected data, a new high-effort feature/function and a high-effort adapted feature/function require respectively at least 15 and 4 hours to be developed by one experienced developer.

Once the Tukutuku variables had been sketched out and explained, the next step was to remove all variables that were not relevant for the DEs, followed by adding to the white board any additional variables (factors) suggested by them. This entire process was documented using digital voice recorders and also text editors. We also documented descriptions and rationale for each factor proposed by the DEs. The factors proposed were indeed influenced by DEs' hunches and insights; however DEs decisions and choices were also very much influenced by their solid previous experience managing Web projects, and estimating development effort.

Next, we identified the possible states that each factor would take. All states were discrete. Whenever a factor represented a measure of effort (e.g. Total effort), we also documented the effort range corresponding to each state, to avoid any future ambiguity. For example, to one of the participating Web companies, 'very low' Total effort corresponded to 4+ to 10 person hours, etc. Once all states were identified and thoroughly documented, it was time to elicit the cause and effect relationships. As a starting point to this task we used a simple medical example from [21] (see Fig. 3).

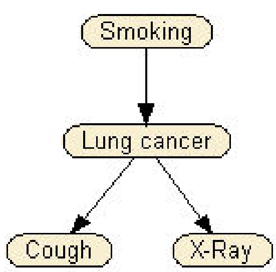


Fig. 3. A small BN model and two CPTs

This example clearly introduces one of the most important points to consider when identifying cause and effect relationships – timeline of events. If smoking is to be a cause of lung cancer, it is important that the cause precedes the effect. This may sound obvious with regard to the example used; however, it is our view that the use of this simple example significantly helped the DEs understand the notion of cause and effect, and how this related to Web effort estimation and the BNs being elicited. Once the cause and effect relationships were identified, we worked on the elicitation of probabilities to quantify each of the cause and effect relationships previously identified. In all four cases, there was an iterative process between the structural development and parameter elicitation steps.

Detailed Model Validation: Both Model walkthrough and Predictive accuracy were used to validate all four Web Effort BN models, where the former was the first type of validation to be employed in all cases. DEs used different scenarios to check whether the node Total_effort would provide the highest probability to the effort state that corresponded to the DE's own suggestion. However, it was also necessary to use data from past projects, for which total effort was known, in order to check the model's calibration. Table 1 details the number of projects used by each company as validation set. In all cases, DEs were asked to use as validation set a range of projects presenting different sizes and levels of complexity, and being representative of the types of projects developed by their Web company.

For each project in a validation set, evidence was entered in the BN model, and the effort range corresponding to the highest probability provided for 'Total Effort' was compared to that project's actual effort. Whenever actual effort did not fall within the effort range associated with the category with the highest probability, there was a mismatch; this meant that some probabilities needed to be adjusted. In order to know which nodes to target first we used a Sensitivity Analysis report, which provided the effect of each parent node upon a given query node. Within our context, the query node was 'Total Effort'.

Whenever probabilities were adjusted, we re-entered the evidence for each of the projects in the validation set that had already been used in the validation step to ensure that the calibration already carried out had not affected. This was done to ensure that each calibration would always be an improved upon the previous one. Once all projects were used to calibrate the model the DEs assumed that the Validation step was complete.

All four BNs have been in production for over a year.

Figs 4 to 7 show the BNs used herein.

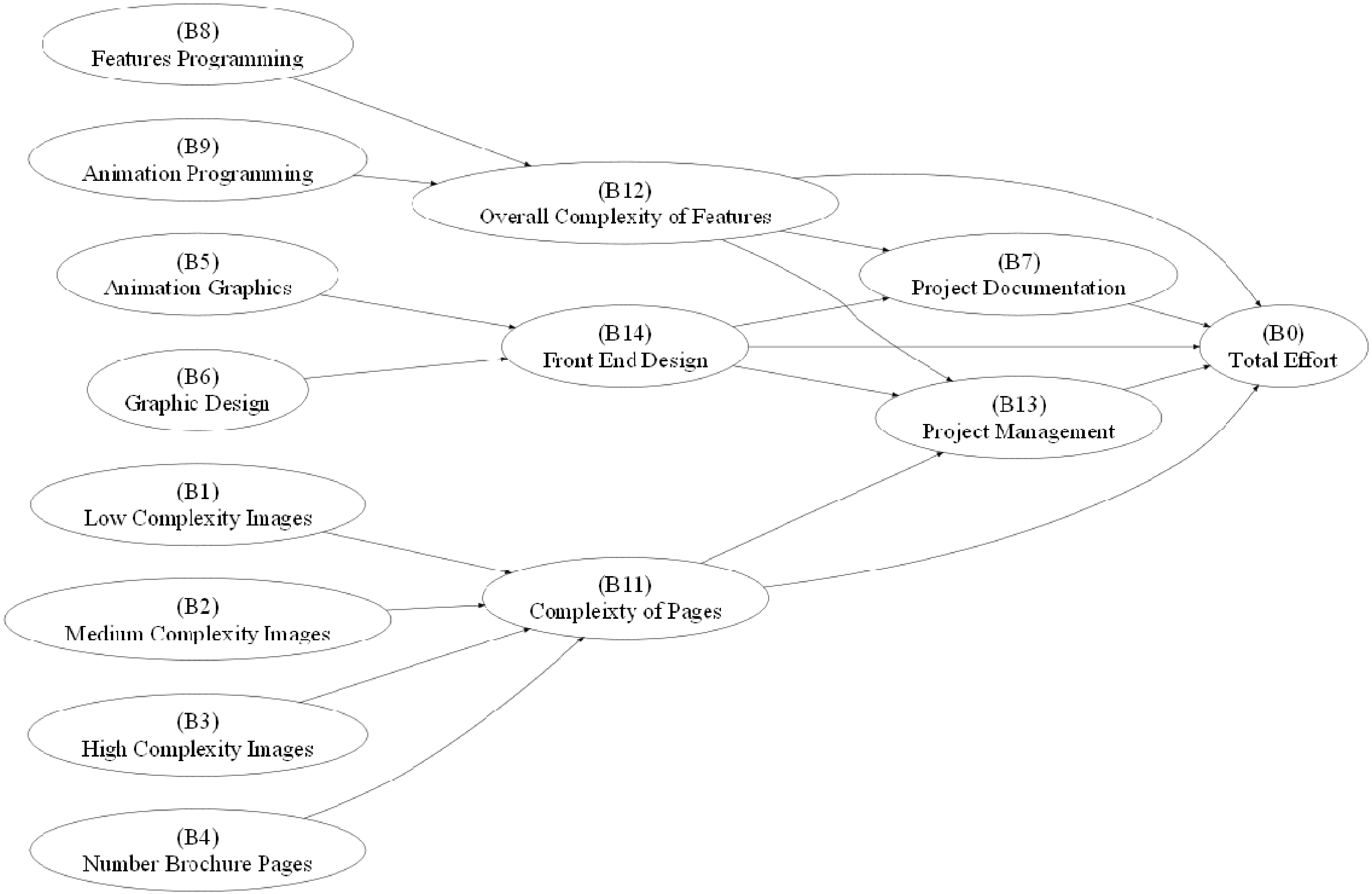


Fig. 4. BN from Web Company B

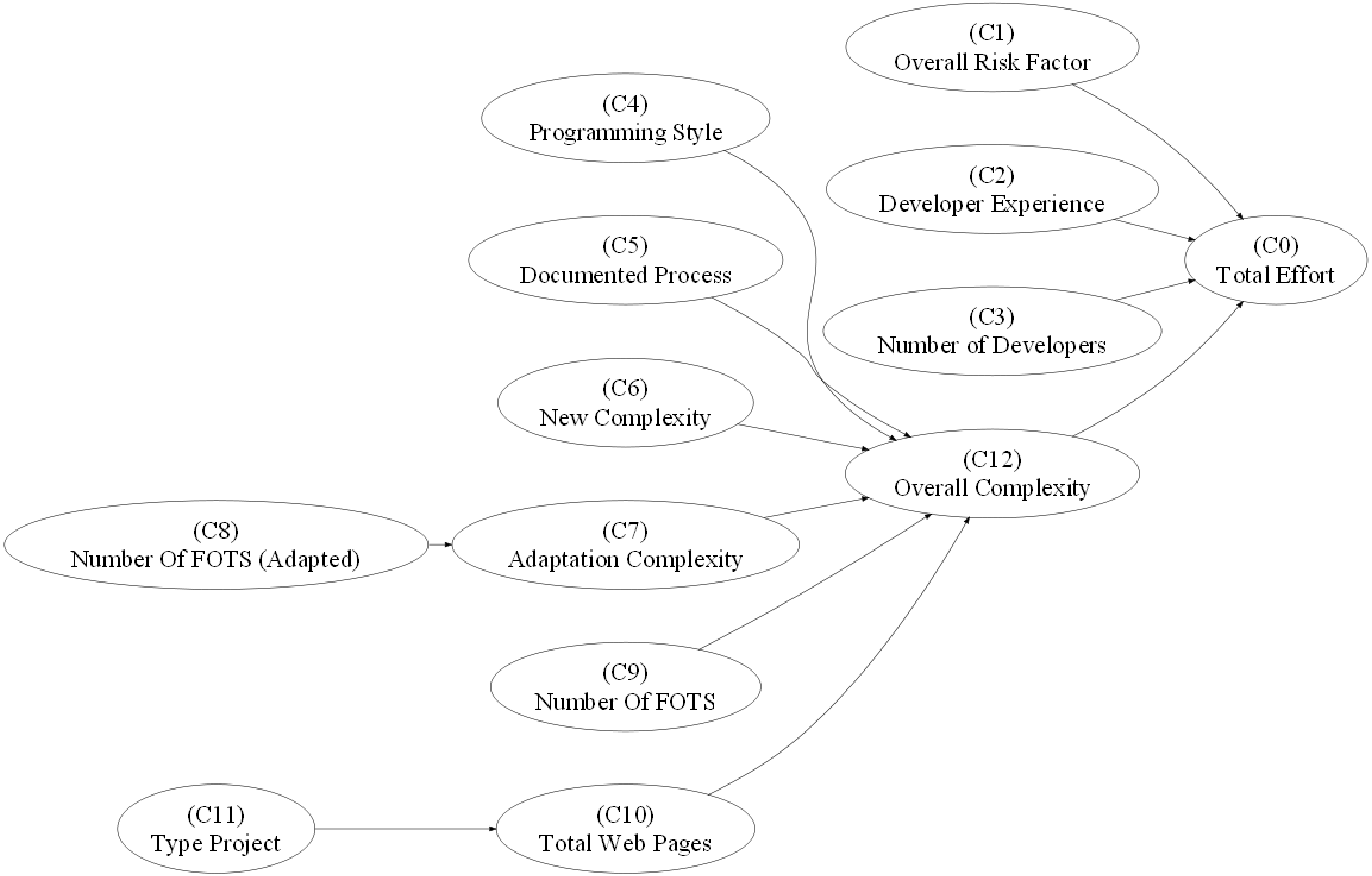


Fig. 5. BN from Web Company C

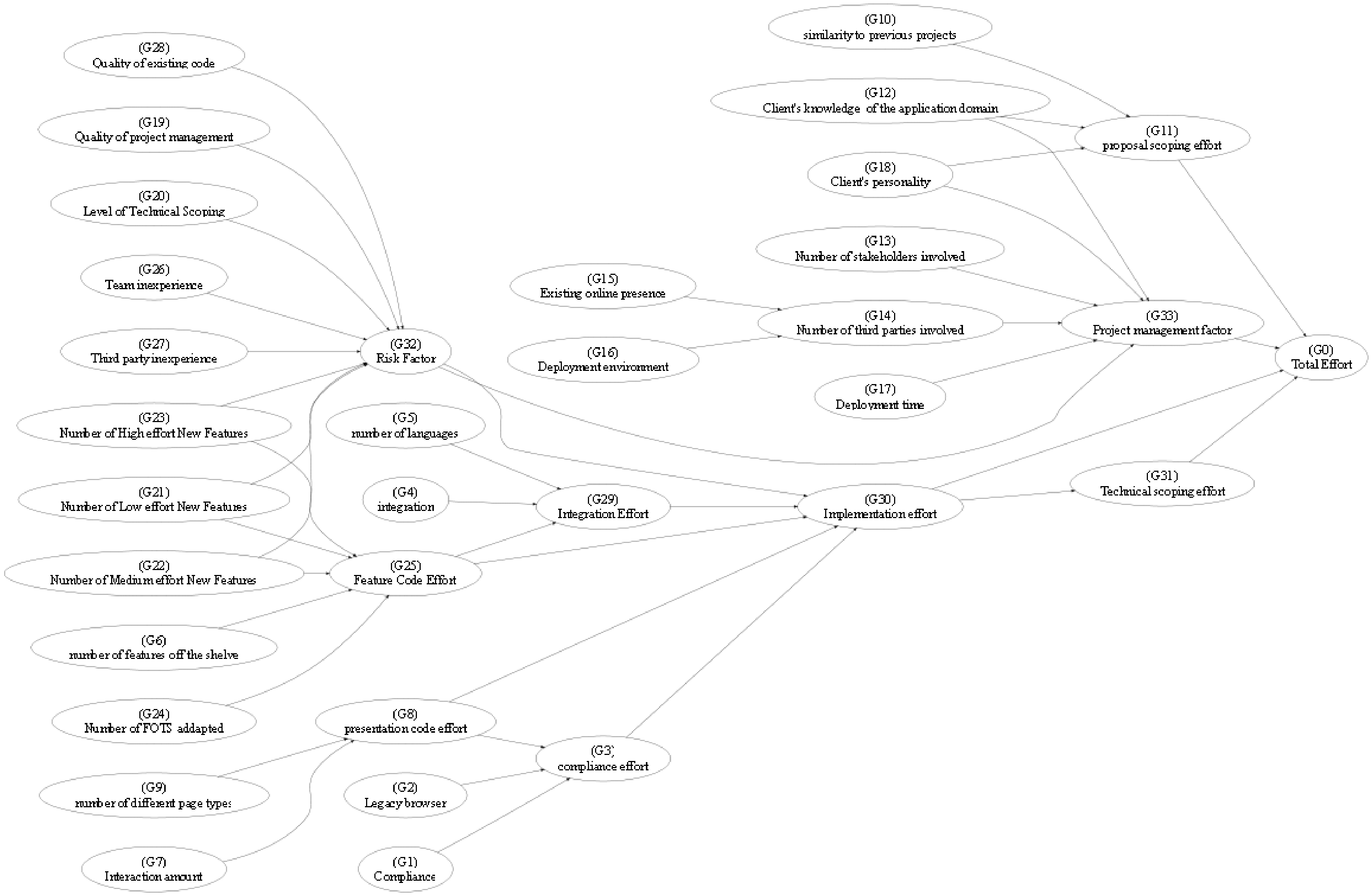


Fig. 6. BN from Web Company G

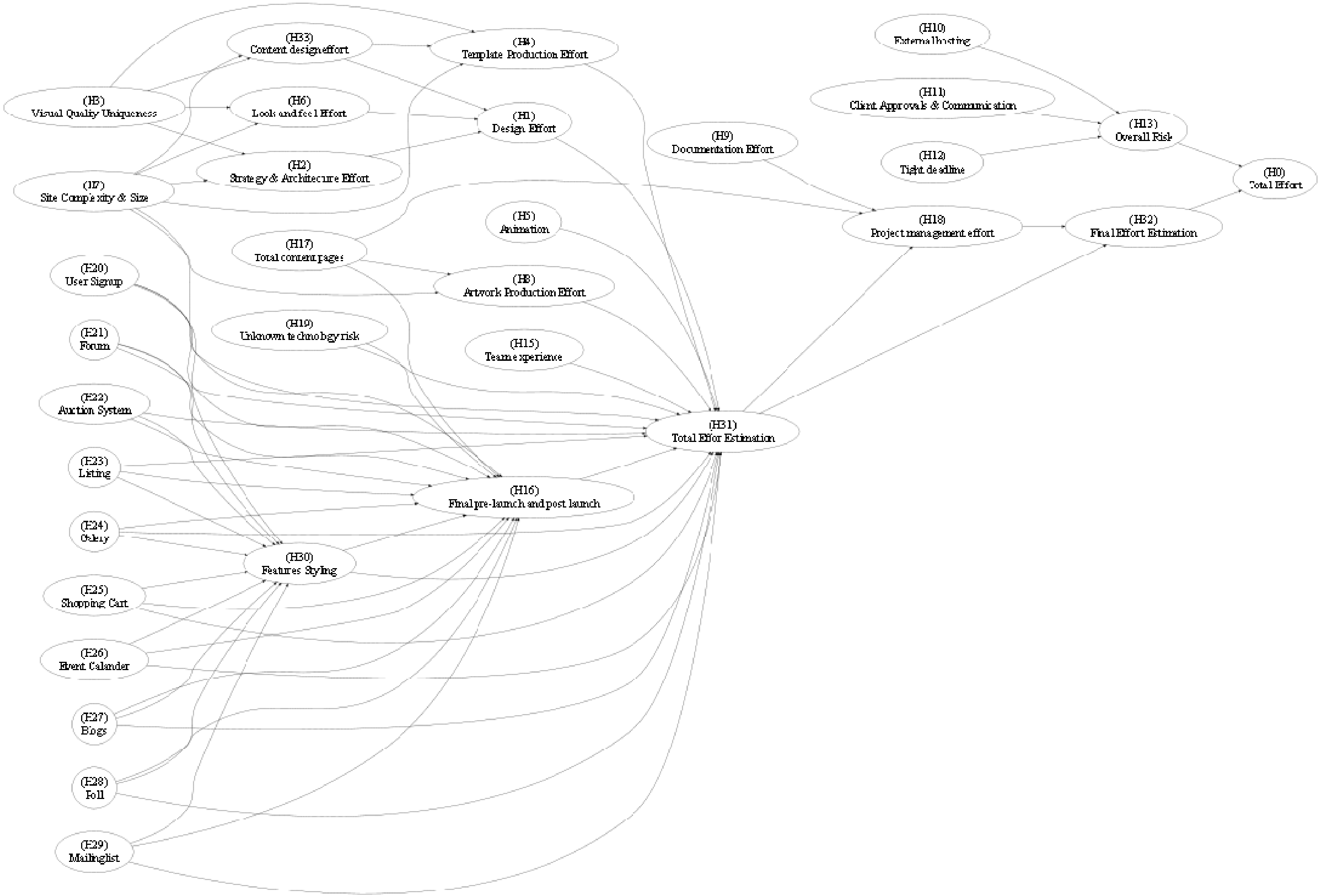


Fig. 7. BN from Web Company H

4 Building an Empirical Generalisation

4.1 Problems Relating to the Aggregation of Typologies[2]

It is often recommended that typologies be constructed through elicitation from different domain experts in order to derive a comprehensive and accurate model [1],[50],[17],[18]. However, it is difficult to combine the beliefs of different experts in a coherent and impartial manner.

In order to arrive at a comprehensive Typology we would need to consult domain experts, many of whom working for different and perhaps competing companies, and thus likely to have a different prospective about the Web development domain. Therefore, the difficulty in combining expert-based typologies increases for the following reasons:

- Identifying Common Variables: Different experts might represent semantically equivalent concepts in their typologies using different variable names (e.g. '*Number of Developers*' vs. '*Project Human Resources*'). Furthermore, experts might use a different number of variables to represent the same concept.
- Conflicting Causal Relations: Variables might have contradictory causal relations according to different experts. Two kinds of causal relation conflicts can occur: the first when there is a causal influence between two variables according to an expert's belief, which is strictly prohibited by another expert's belief. The other type of conflict is the occurrence of cycles (which is ruled out within the context of this work in order to keep the resulting aggregated typology consistent with all the four individual typologies being used as input, which were all Directed Acyclic Graphs (DAGs)).
- Collaboration Constraints: One feasible way to construct a generic model for Web effort estimation is to elicit a single typology from a group of domain experts from a representative sample of Web development companies. This would need to be done in stages, and such approach might work well with small groups of domain experts but will likely to be impractical when additional typologies are included in the unified model. However, within the context of this research, any form of collaboration between domain experts is not feasible because all of the participating companies compete in the same market. This means that, by collaborating with other experts, they would be forced to share sensitive business information that they are not willing to disclose.

Therefore, it is vital to apply a methodology for combining different expert-elicited typologies that solves the difficulties abovementioned. In this work we use the same methodology employed in our previous work [1][2], which solves many of the affiliated challenges in combining expert-elicited models that have not been sufficiently addressed in prior work. A detailed explanation of this methodology is given in [1].

4.2 Problems Relating to the Aggregation of Typologies

We proposed a qualitative methodology that pragmatically addresses the shortcomings of previous studies as follows:

1. Introducing a mapping scheme, i.e., a way to identify similar existing variables in the participating companies' typologies.

[2] Typology here is equivalent to the qualitative component of a BN, i.e., its causal map

2. Instead of using a simple union/intersection, which can only include a common node or edge exactly once, we attempt to aggregate the causal structures. By aggregation, we imply that all edges and nodes in the original typologies are preserved. As more typologies are aggregated, the most common variables and causal links emerge, thereby simulating in our view a form of consensus between the different companies' typologies.

We termed the resultant aggregated graph as a Causal Structure Aggregation Model (CSAM). Strictly speaking, it is not a unified model, but it is a tool for discovering a consensual model. A CSAM is a graph that represents the cumulative addition of individual typologies according to a node mapping scheme. The aim of a structure aggregation model is to identify causal commonalties between independently developed typologies that share the same domain. Consider the three typologies presented in Figure 3. All are used to estimate the total effort required to develop a Web application. Since they all share the same domain, it is possible to assume that the nodes in two different typologies portray the same factor. For example, nodes A1, B1, and C1 all model the same factor - *the number of developers required to develop a Web application*, and therefore, it is possible to map those three nodes into a single factor.

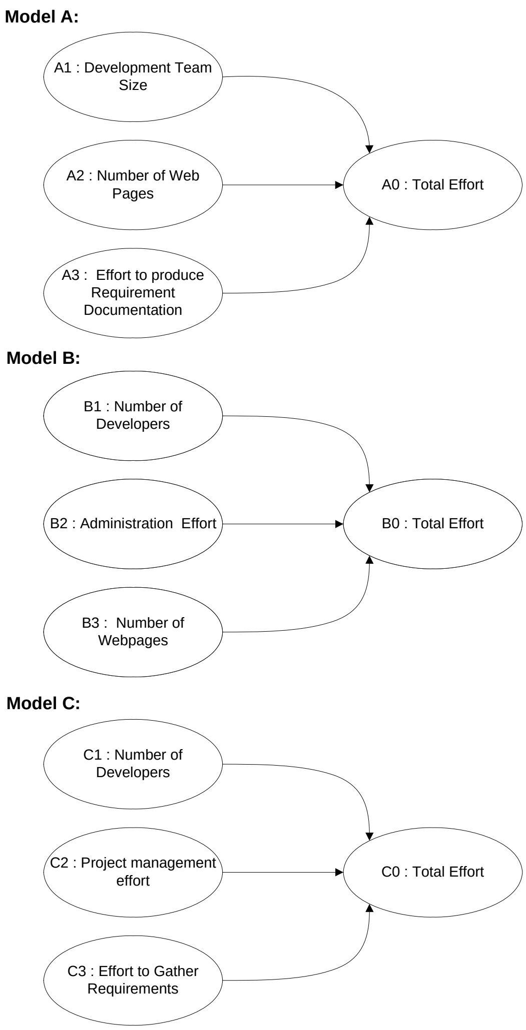


Fig. 8. Three Basic Examples of Models

Some nodes are more subjective in their definition, e.g., nodes B2 and C2 both attempt to model the effort required to develop a Web application, but the exact details of how to measure this effort might vary between the two companies. However, because both typologies share the same domain, both B2 and C2 are likely to portray the same underlying concept. By performing this type of mapping between the three typologies, we can produce the CSAM presented in Figure 9.
The left partition of a CSAM's node represents a factor of interest, while the right partition contains a list of nodes from the original typologies that map to this factor. All the causal links from the original typologies are preserved in the CSAM, i.e., if there is a link between two nodes in one of the original models (for example from A2 to A0), then in the CSAM there must be an edge from every node that contains A2 in its mapping to every node that contains A0 in its mapping. The numbers attached to the edges in the CSAM represent the cardinality of their mapping. For example, the edge from node (1) to node (0) has a cardinality of three; this is because there are three original edges that map to it: A1 to A0, B1 to B0, and C1 to C0. The cardinality of a node is the number of 'original' nodes that it maps to (i.e. the number of nodes listed in its right partition). The example in Figure 9 has a simple one-to-one mapping between the CSAM factors and the nodes from the example typologies. It is possible to have a many-to-many mapping to resolve more ambiguous situations.

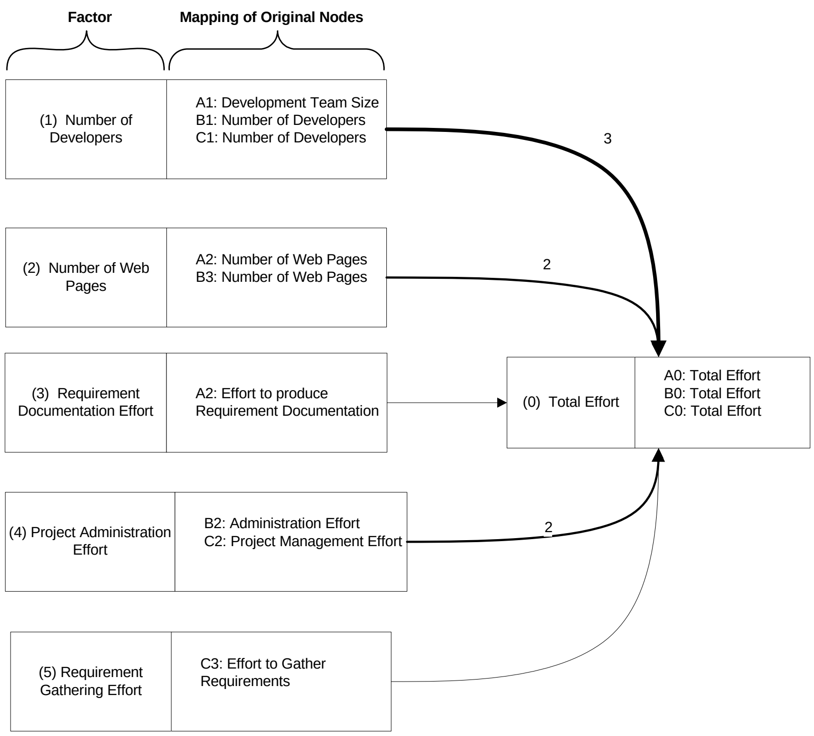


Fig. 9. CSAM model

4.3 Methodology

The main goal of this work is to build a CSAM to obtain an empirical generalization of four different causal process form type of theories for Web effort estimation. The methodology used to combine these theories comprised a six-step process (detailed below) combining both linear and iterative approaches (see Fig. 10).

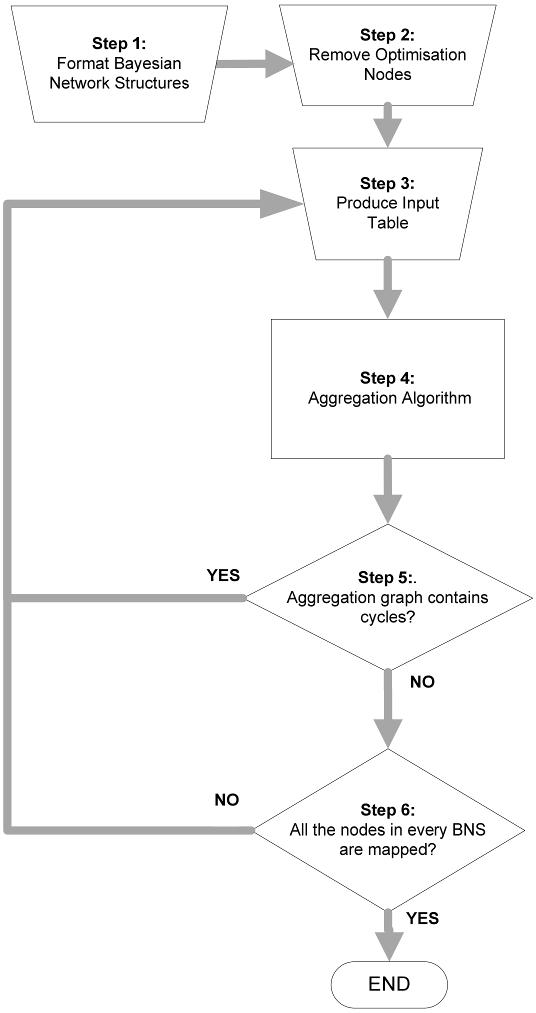


Fig. 10. Process Flow Diagram for Producing a CSAM

1) Formatting of the companies' models

The companies' typologies were first formatted so they could be handled by the aggregation algorithm (step 4). The formatting consisted of the following steps:
Each node in every typology was given a unique Identifier. The identifiers chosen for our research represented a concatenation between a company's model identifier and a unique natural number (a number only valid within the context of a single typology). Each typology was represented in a parseable format, where the format chosen herein was CSV (Comma Separated Values).
The choice relating to the identifiers' representation and parsing format to use was informed by the tool implemented to help with this aggregation process.

2) Removal of optimisation nodes

Optimisation nodes are intermediate nodes that were inserted into a causal structure to partition large CPTs in order to reduce their probability elicitation effort. In general, such

nodes are not part of the original model elicited with the domain experts; rather, they are suggested by the Knowledge Engineer, and approved by the experts. The purpose of our CSAM was to only aggregate the factors and causal relationships originally modelled by the experts, and as such, the inclusion of optimisation nodes was deemed inappropriate.
Optimisation nodes were first identified from the documentation available for each of the companies' theories. To remove an optimisation node we connected all of its incoming edges (coming from its parent nodes) directly to all of its child nodes, followed by the removal of this optimisation node and all of its outgoing edges (see Fig. 11):
During this operation BNs' existing graph rules must always hold. For example, only a single edge could have the same source and destination nodes; therefore, if the removal of an optimisation node resulted in adding an edge between two nodes that were already directly linked, then the resultant edge had to be discarded.

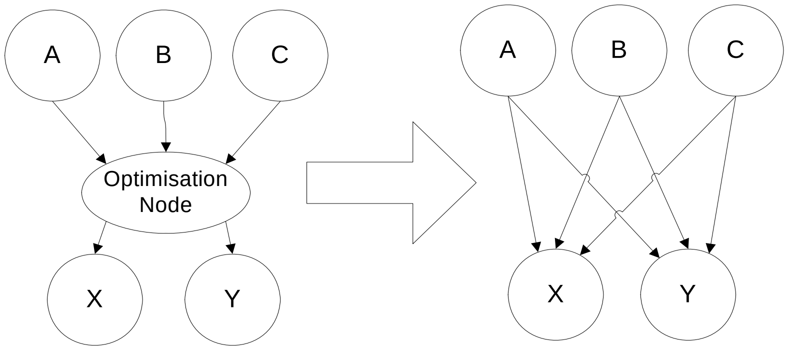


Fig. 11. Removing an Optimization node

3) Creation of an Input Table

Each node in our CSAM corresponded to a semantically equivalent node originating from one of more of the typologies used as input. Sometimes different typologies would contain the same node however named differently; when carrying out the mapping (as detailed below) we checked for the semantic equivalence between nodes across typologies. These mappings were documented using a Table, where each row was used to map a CSAM node to all the other semantically equivalent nodes originating from models. The table's first column represented a CSAM node (factor), identified by a unique ID; the remaining columns contained node identifiers associated with the nodes contained in the input typologies.
Given a company's input typology, the first node to be mapped was the most-posterior node, which within our context always happened to be the *Total Effort*. We chose this node because it was part of all the participating companies' typologies, and therefore we believed it to be the easiest node to identify and map. Once *Total Effort* was mapped, the remaining nodes were mapped according to the following steps:

1. Selection of a node (factor) from a company's typology that had not yet been mapped.
2. Identification of the contextual meaning of the factor selected in (1), which usually involved interpreting the underlying concept that the DE employed when that factor was elicited. We first identified the units and quantification used to measure the factor, followed by looking at the supporting documentation from the elicitation sessions, which contained examples, additional commentary about the DEs' beliefs,

and digitally recorded voice records documenting all the elicitation process. In the rare cases where a factor's contextual meaning was still ambiguous, the DEs were contacted for clarification.

3. Attempt to map the factor identified in (1) (f) to a factor, or set of factors, already present in our CSAM. Whenever there was no corresponding factor(s) clearly mapping to f, we created a new factor(s) within our CSAM to match that given factor f.

There were no strict rules as to whether an original node was mapped to one or more factors within our CSAM; however, we always aimed to keep as much of the original context as possible through the mapping. Thus the reason why our methodology is iterative and not linear is because mappings often change as new factors are created and old ones are revised.

In order to minimise the effort of constantly changing the mappings as the aggregation map was populated, we decided to map the original nodes in different iterations rather than mapping all nodes at once. This gave us the opportunity to run the aggregation algorithm (see step 4 below) and generate the CSAM several times, containing incomplete aggregation models, and then to look for faults and inconsistencies (e.g. cycles). The first iteration involved mapping every prior node from all the companies' typologies. The second iteration involved mapping all the nodes from all the companies' typologies that were directly pointed to by all prior nodes, and so on until the most posterior (the *Total Effort* Node) was reached.

4) Aggregation algorithm

The Table prepared in step 3 was used as input to an aggregation algorithm that produced a graphical representation of the CSAM. The algorithm worked by first merging the prior nodes according to the mapping specified in the Table, and continuing until all nodes in all the companies' typologies were processed [1].

Whenever the Table from Step 3 did not include mappings for some of the nodes in the inputted typologies, then these nodes were represented in the CSAM by *placeholder* nodes. The purpose of the placeholder node was so that we were aware of which nodes still required mapping in the next iteration of this process (see Step 6).

5) Check if the Aggregation Graph contains cycles

The aggregation algorithm allowed for the occurrence of cycles since it simply followed what was documented in the Table used as input. Therefore, when the generated CSAM graph contained cycles, the input Table needed to be modified so that all of the documented cycles were broken. Cycles could be broken by changing the mapping of one or more nodes that made up the cycle, which could be achieved by either removing or adding factors to the input Table. However, because all the companies' typologies were independent of one another and yet shared the same domain, it is theoretically possible, in theory, to have cycles occurring that may not be resolved. This would occur whenever nodes in their original typologies did not form cycles, but ended up contributing to a cycle in the CSAM due to conflicting contexts.

6) Check if all nodes are mapped

The final step in the process was to check whether every node (except for optimisation nodes) in all the companies' typologies had been mapped in the CSAM. For this we looked for the existence of placeholder nodes in the CSAM outputted by the algorithm. If found, we mapped the map's nodes identified by the placeholder nodes by referring back to Step 3; conversely, if there were no placeholder nodes, we considered that the CSAM was complete according to our mapping.

5 Results

In this section, we present our results from aggregating four independently elicited single-company BNs. These elicited models varied in their sizes, as previously summarised in Table 1.

The CSAM[3] resulting from our 6-step methodology (presented in Section 4) enabled us to identify common patterns, in terms of variables (factors) and causal relations, shared amongst the four independent causal process form type of theories (single-company BNs). This CSAM presented 70 nodes in total, encompassing all the factors identified by all four participating companies via their BNs. This empirical generalisation, which is characterised by a combined list of factors, brought us one step closer to determining all the causal factors in our target domain (Web development effort estimation) that are significant for Web effort estimation, and therefore closer to a unified causal process form type of theory for Web effort estimation. Table 3 lists the Factors[4] in our CSAM and their cardinality, which corresponds to the number of input BNs that contained that factor. Therefore, a factor's cardinality is an indication of how common this factor was as a predictor amongst the four participating companies.

The most common factor in our CSAM, presenting the highest cardinality on the list, was the '*Project Management Effort'*. This is in our view a very interesting result as it suggests that the companies that participated in this research present some level of maturity in their processes, and consider that managing projects effectively is a very important aspect of delivering applications on time and within budget.

[3] The resultant CSAM is available here: http://www.cs.auckland.ac.nz/~emilia/Theory/CSAM.pdf

[4] A description of all CSAM factors is given here: http://www.cs.auckland.ac.nz/~emilia/Theory/Factors.pdf

Table 3. List of CSAM Factors and their cardinality

Node ID	Label	DataType	Cardinality
0	Total Effort	person hours	4
5	Average Project Team Experience with Technology	years	3
8	Client Personality Difficulty	UD (e.g. Low, Medium, High; good, normal, bad)	3
66	Effort to Program Features	person hours	3
59	Project Management Effort	person hours	3
1	Adaptation Effort of Features off the shelf	person hours	2
13	Development Effort of New Features	person hours	2
17	Effort producing Animations using Software	person hours	2
18	Effort programming Animations	person hours	2
56	Effort Template Look & Feel	person hours	2
64	Effort to Develop User Interface	person hours	2
67	Effort to Implement the Web application	person hours	2
61	Effort to Produce Requirements Documentation	person hours	2
63	Effort to Produce Template Mock-up	person hours	2
60	Effort to Produce Web Pages	person hours	2
74	How much Technical planning	UD (e.g. low, normal, high)	2
21	Number of Features off the shelf	Integer	2
22	Number of Features off the shelf Adapted	Integer	2
34	Number of New Web Pages	integer	2
52	Web Company's Hosting Control	UD (e.g. client in-house, shared, dedicated, in-house)	2
87	Project Risk Factor	UD (e.g. low, medium, high)	2
106	Effort Production testing	person hours	2
107	Effort Post-release testing	person hours	2
49	Amount of text per Application	UD (e.g. low, medium, High)	1
72	Client Application Domain Literacy	UD (e.g. low, medium, High)	1
70	Client's Existing online presence	UD (e.g. small, extensive, none)	1

75	Deployment time	UD (e.g. short, normal)	1
14	Development Process Model	UD (e.g. conventional, waterfall, extreme)	1
15	Development team size	integer	1
16	Effort Images Manipulation	person hours	1
68	Effort to Integrate New and Reused Features	person hours	1
19	Is Development Process Documented?	Yes/No	1
23	Number of Features requiring High effort to create	Integer	1
25	Number of Features requiring Low effort to create	Integer	1
69	Number of Features requiring Medium effort to create	Integer	1
29	Number of Images requiring High effort to manipulate	integer	1
30	Number of Images requiring Low effort to manipulate	integer	1
31	Number of Images requiring Medium effort to manipulate	integer	1
51	Number of Key Client's people	Integer	1
35	Number of Reused Web Pages	integer	1
71	Number of third parties involved	Integer (e.g. sub-contractors, printing, SMS gateways, hosting providers, domain registration, payment providers)	1
37	Number of Web Page Templates	integer	1
77	Quality of In-house Existing Code	UD (e.g. low, normal, high)	1
76	Quality of Project Management	UD (e.g. abysmal, low, normal, high)	1
50	Quality of Third Party Deliverables	UD (e.g. low, high)	1
73	Technical planning effort	person hours	1
46	Type of Project	UD (e.g. New, Enhancement)	1
79	Level of Usability	UD (e.g. low, medium, high)	1
80	Similarity to Previous Projects	UD (similarity of domain/functionality/design; e.g.: low, medium, high)	1
82	Legacy browser support	UD (e.g. yes, no)	1
83	Effort to Implement Accessibility	person hours	1
84	Level of Integration between Features	UD (e.g. low, medium, high)	1
85	Number of Natural Languages Used	integer	1
86	Total third party inexperience	UD (e.g. low, medium, high)	1
88	unknown technology risk	UD (boolean)	1

89	Forum Feature	UD (boolean)	1
90	User sign up feature	UD (boolean)	1
91	Auction system feature	UD (boolean)	1
92	types of Listing features	UD catagories	1
93	Gallery feature (number of controls)	UD (number of widgets)	1
94	shopping cart feature	UD (boolean)	1
95	event calander feature	UD (boolean)	1
96	Number of Blogs	Integer	1
97	Number of Poll	integer	1
98	Mailling List feature	UD (boolean)	1
99	Effort to produce user documentation	person hours	1
100	Tight schedule	Bollean	1
103	Template design uniqueness	UD (e.g. Template Standard, Template High, Custom-medium, Custom-high	1
104	Effort to Design Content	person hours	1
105	Effort to Implement the Template	person hours	1

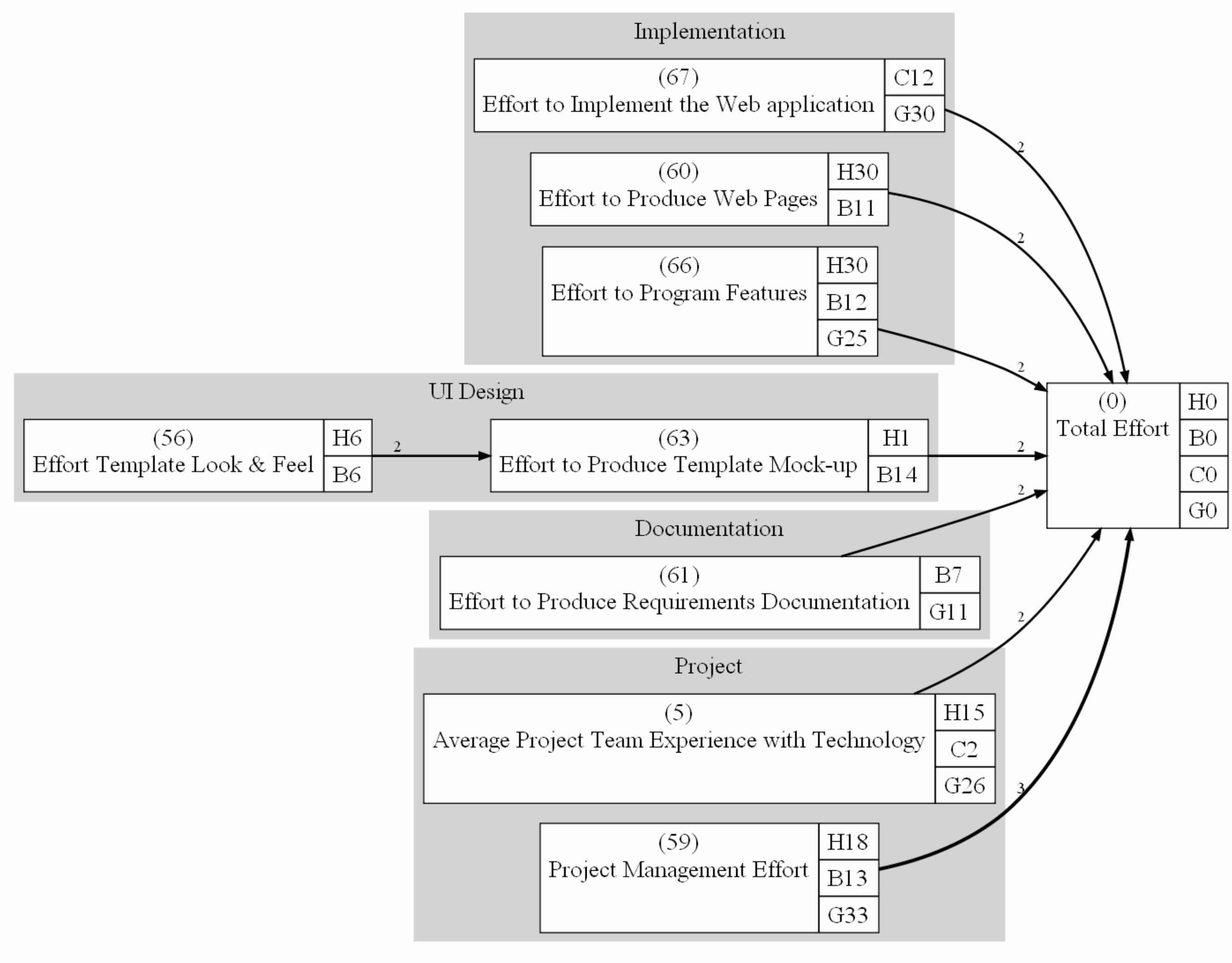
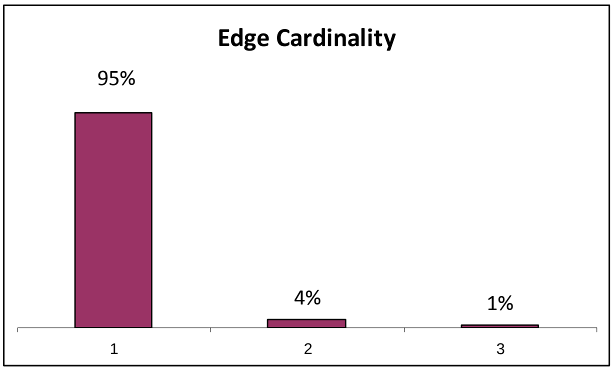


Fig. 12. A sub-graph of the resultant CSAM with all edges having cardinality greater than two

Fig. 12 shows a sub-graph of the resultant CSAM with all edges having cardinality greater than two. Factors were grouped into higher level categories (grey boxes) in order to aid readers understand it. These categories are: Implementation-oriented factors, UI Design-oriented factors, Documentation-oriented factors, and Project-oriented factors. Note that most factors measure the effort required to carry out a particular task (e.g. manage projects, produce requirements documentation). This figure can be described as an aggregated intersection of all the causal edges in the inputted Typologies. The higher the weight value of an edge the more common the causal relation is. This figure is therefore very useful as it indicates likely relationships that exist between factors within the Web development domain. We believe that as we further aggregate other typologies to our resultant CSAM, a more informative and decisive consensus will emerge relating to its empirical generalisation, thus also strengthening the external validity of this model. In other words, a CSAM is a maturing model, providing further certainty as further typologies are aggregated.

Note that when we consider the four BNs that were aggregated (see Figs. 4 to 7), they present numerous effort-related factors, suggesting that their factors and corresponding causal relationships also seem to relate to their Web effort estimation workflow. This is in our view quite an interesting aspect as it was a recurring theme in three of the four BNs employed herein. Another observation is that factors that perhaps may have higher importance in conventional software development also seemed to be common amongst the

participating companies; for example, factors related to requirements engineering and documentation.
Figure 13 shows the proportion of factors according to their equivalent nodal cardinality. We can see that approximately 67% of all factors appeared in only a single company's BN, and 26% of factors were common to at least two BNs. The percentage of nodes decreased as the cardinality increased, suggesting that the total number of factors available in the target domain significantly outnumbered the factors being considered by individual companies. Likewise, the percentage of causal edges also rapidly decreases with respect to edge cardinality, which suggests that there are many causal relationships not considered by individual companies.



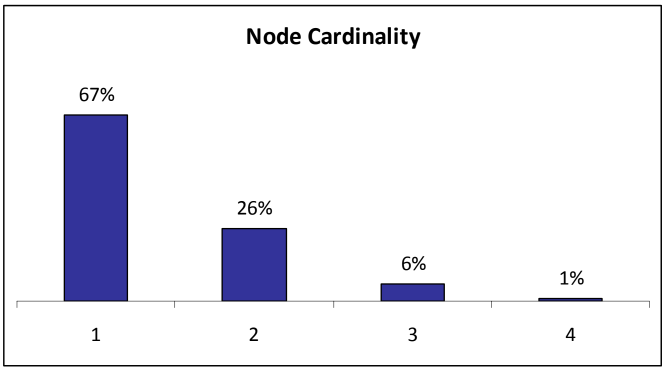


Fig. 13. Distribution of node and edge cardinalities in our CSAM

There were 158 causal edges our CSAM. We were able to determine the most common causal relations by selecting all the matched causal relations in the CSAM, that is, all causal edges with a cardinality of two or more. Our results showed that 26% of all causal relations were shared between at least two BNs. The most prevalent causal relationship was between factors '*Project Management Effort*' and '*Total Effort*' whereby 66.6% of the participating companies included such relationships in their causal maps. Other common causal relationships identified were:

- Relationship from '*Effort to Implement the Web Application*' directly influencing *'Total Effort'*.
- Relationship from *'Effort to Produce Web Pages*' directly influencing *'Total Effort'.*
- Relationship from *'Effort to Program Features'* directly influencing *'Total Effort'.*
- Relationship from *'Effort to Produce Template Mock-up'* directly influencing *'Total Effort'.*
- Relationship from *'Effort to Produce Requirements Documentation'* directly influencing *'Total Effort'.*
- Relationship from *'Average Project Team Experience with Technology'* directly influencing *'Total Effort'.*

Each of the six causal relations listed above appeared in 50% of the companies' BNs. Edges with higher cardinality tended to be closer to the most posterior node (*'Total Effort'*). Fig. 14 shows a falling trend in the mean and median average distances to the Total Effort node. An average mean distance = 0.95 for edges with cardinality of 1, and mean distance = 0.14 for edges with cardinality of 2. This is in our view an important outcome because 'effort' is what all the BNs used in this research aim to predict; it is therefore advantageous to know which factors were likely to have a direct effect upon effort, since this would be the focal point of any future consensus-based BN.

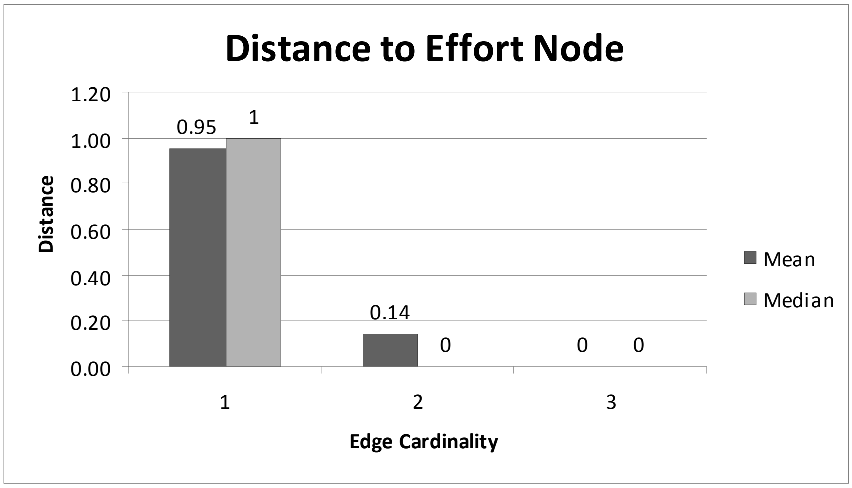


Fig. 14. Average Distance by edge cardinality to Total Effort node

6 Threats to Validity

There are a few threats to the validity of our work. One is the mapping of original nodes (i.e. creating the aggregation map as part of the third step in our methodology, as detailed in Section 4). The mapping was performed by the researchers (i.e. knowledge engineers), not the domain experts; therefore, there is always the possibility of bias being introduced. However, it is important to note that many steps were undertaken to mitigate this risk. All mappings were based on an extensive documentation provided by the experts, and for cases where there was still ambiguity, the experts were contacted directly for further clarification.

Another threat is that our methodology does not in any way guarantee that the final CSAM is free of cycles. Although for the four company typologies, all potential cycles were resolved by further investigation and remapping; this might not always be the case. It is always possible to have intrinsically contradictory typologies, rendering it impossible to resolve cycles unless at least one edge is omitted from the CSAM.

Finally, for the CSAM to be fully comprehensive in terms of domain factors, it is necessary to aggregate a large number of typologies. For our case, the aggregation of four typologies is not enough to represent all factors and causal relations that impact effort estimation in the Web development domain. However, we note that the resultant CSAM is a maturing empirical generalisation, and we plan to aggregate further typologies as part of our future work.

7 Conclusions

The aim of this paper was to obtain an empirical generalisation comprising important factors for Web effort estimation and their cause and effect relationships by aggregating the Typologies from four single-company Web effort estimation BNs. We also contend that each of the BNs used here represents a causal process form type of theories for Web effort estimation, where each of these theories was previously built using a hybrid approach to theory construction, based on existing knowledge elicited from several domain experts, data on past completed Web projects, and a technique that enables the modeling of causal relationships and their uncertainty.

To build such an aggregated model presents numerous challenges, namely identifying common variables, resolving causal relation conflicts, and company collaboration constraints. However, we believe that one can overcome some of these challenges by applying an aggregation process that can yield the most common patterns shared between single-company typologies [2]. Our proposal for building an aggregated map was based on an earlier proposition by Sagrado et al. [57], which attempted to combine BNs' typologies using intersection/union of DAGs forming a consensus causal structure. Our proposal improved upon this proposition in two ways: first by introducing a mapping mechanism for grouping related variables from different single-company typologies, and secondly by using an aggregation of nodes and edges instead of a simple union/intersection, thus preserving all edges and nodes from the original typologies. We termed the aggregated causal map as a Causal Structure Aggregation Model (CSAM), and its chief rationale was to identify structural commonalities (common factors and causal relations) found in the original typologies.

We have constructed a CSAM using four expert-driven single-company typologies (part of single-company BNs), all of which elicited from local Web development companies in Auckland, NZ. This CSAM contained 70 factors and 158 causal edges.

The resultant CSAM revealed the following patterns: i) 33% of the CSAM factors were shared between at least 2 single-company typologies; ii) The most common factor was '*Project Management Effort*'; iii) The proportion of nodes rapidly decreased as cardinality increased, implying that the total number of factors relevant in the Web effort estimation domain significantly outnumbers the number of factors being considered by individual companies; iv) 5% of all causal relations found in the CSAM were shared between at least two single-company maps; v) The most common causal relationship in our CSAM was between factors '*Project Management Effort*' and '*Total Effort*', included in 66.6% of the single-company maps; vi) Six other common causal relationships which were evident were: '*Effort to Implement the Web Application*', '*Effort to Produce Web Pages*', '*Effort to Program Features*', '*Effort to Produce Template Mock-up*', '*Effort to Produce Requirements Documentation*', and '*Average Project Team Experience with Technology*', all of which directly influenced '*Total Effort*'; vii) Edges with higher cardinality tended to be closer to the most posterior node, suggesting that most factors influenced total effort directly.

The abovementioned points show that even with a small number of companies we can already see reasonable commonality in terms of factors and causality. The CSAM is a maturing model, which means that as more typologies are aggregated; the more common factors and causal links will emerge, hence providing an improved consensus on its empirical generalisation. The aggregation process presented herein can be used to aggregate other typologies. In addition, to our knowledge this is the first time that a study in either Web or

Software Engineering describes argues that BNs and their documentation in fact represent causal process form type of theories, and an empirical generalisation via the aggregation of several single-company typologies. Our future work involves the aggregation of other expert-driven single-company typologies, and later the proposal of a general causal process form type of theory for Web effort estimation.

10 Acknowledgements

We thank Professor Jorgensen for their comments on an earlier version of this paper, and also all the participating companies in this research.

References

1 Baker, S., Towards the Construction of Large Bayesian Networks for Web Cost Estimation, MSc thesis, in Department of Computer Science Auckland: University of Auckland, 2009.

2 Baker, S. and Mendes, E. Aggregating Expert-driven Causal Maps for Web Effort Estimation, Proceedings of the Advances in Software Engineering (ASEA) Conference: Communications in Computer and Information Science, Volume 117, pp. 264-282, DOI: 10.1007/978-3-642-17578-7_27

3 Brown, B. "Delphi process: A methodology used for the elicitation of opinions of experts," Santa Monica, CA, Rand Corporation, 1968.

4 Baresi, L., S. Morasca, and P. Paolini. An empirical study on the design effort for Web applications. Proceedings of WISE 2002, 345-354, 2002.

5 Baresi, L., S. Morasca, and P. Paolini. Estimating the design effort for Web applications. Proceedings of Metrics 2003, 62-72, 2003.

6 Castillo, E., J. M. Gutiérrez, and A. S. Hadi, "Combining multiple directed graphical representations into a single probabilistic model," in Actas de la Séptima Conferencia Espanola para la Inteligencia Artificial, CAEPIA, 1997, pp. 645–652.

7 Corazza, A., S. Di Martino, F. Ferrucci, C. Gravino, E. Mendes. Applying Support Vector Regression for Web Effort Estimation using a Cross-Company Dataset, Proceedings of the ACM/IEEE Symposium on Empirical Software Measurement and Metrics, pp. 191-202, 2009.

8 Corazza, A., S. Di Martino, F. Ferrucci, C. Gravino, E. Mendes. Using Support Vector Regression for Web Development Effort Estimation, Proceedings of the 4th International Conference on Software Process and Product Measurement (MENSURA'09), pp. 255-271, 2009.

9 Corazza, A., S. Di Martino, F. Ferrucci, C. Gravino, F. Sarro, E. Mendes. How the Choice of the Fitness Function Impacts on the Use of Genetic Programming for Software Development Effort Estimation? Proceedings of PROMISE 2010, Best paper award.

10 Dubin, R. Theory Building, The Free Press, NY, 1969.

11 Das, B.: Generating Conditional Probabilities for Bayesian Networks: Easing the Knowledge Acquisition Problem, arxiv.org/pdf/cs/0411034v1 (accessed in 2008), 2004.

12 Druzdzel, M.J., and van der Gaag, L.C.: Building Probabilistic Networks: Where Do the Numbers Come From?, IEEE Trans. on Knowledge and Data Engineering, 12(4), 481-486, 2000.

13 Fenton, N., Marsh, W., Neil, M., Cates, P., Forey, S. and Tailor, M.: Making Resource Decisions for Software Projects, Proc. ICSE'04, pp. 397-406, 2004.
14 Ferrucci, F., C. Gravino, R. Oliveto, F. Sarro, E. Mendes, Investigating Tabu Search for Web Effort Estimation, Proceedings of Euromicro SEAA 2010 conference, 2010.
15 Fewster, R., and E. Mendes. Empirical Evaluation and Prediction of Web Applications' Development Effort, Proc. EASE'00, 2000.
16 Fewster, R., and E. Mendes. Measurement, Prediction and Risk Analysis for Web Applications. Proceedings of IEEE Metrics Symposium, 338 – 348, 2001.
17 Fink, A., J. Kosecoff, M. Chassin, and R. H. Brook, "Consensus methods: characteristics and guidelines for use," American Journal of Public Health, vol. 74, p. 979, 1984.
18 Flesch, I., P. Lucas, J. A. Gamez, and A. Salmeron, "Markov Equivalence in Bayesian Networks," 2007.
19 Gellersen, H., Wicke, R., and Gaedke, M. WebComposition: an object-oriented support system for the Web engineering lifecycle, Computer Networks and ISDN Systems, Volume 29, Issues 8-13, Pages 865-1553 (September 1997) Papers from the Sixth International World Wide Web Conference, Pages 1429-1437,1996.
20 Hu, X.-x., H. Wang, and S. Wang, "Using Expert's Knowledge to Build Bayesian Networks," Proceedings of the 2007 International Conference on Computational Intelligence and Security Workshops, pp. 220-223, 2007.
21 Jensen, F.V. An Introduction to Bayesian Networks, UCL Press, London, 1996.
22 Kiran, C., The Web Development Industry Is Expected To Grow Over 20 By 2010, http://www.articler.com/23205/The-Web-Development-Industry-Is-Expected-To-Grow-Over-20-By-2010.html (accessed in March 2011)
23 Mahoney, S.M., and Laskey, K.B.: Network Engineering for Complex Belief Networks, Proc. Twelfth Annual Conference on Uncertainty in Artificial Intelligence, pp. 389-396, 1996.
24 Mangia, L., and R. Paiano. MMWA: A Software Sizing Model for Web Applications, Proc. Fourth International Conference on Web Information Systems Engineering, 53-63, 2003.
25 Martino, S. Di, F. Ferrucci, C. Gravino, and E. Mendes. Comparing Size Measures for Predicting Web Application Development Effort: A Case Study, Proceedings ESEM'07, 2007.
26 Mendes, E. "Predicting Web Development Effort Using a Bayesian Network," in Proceedings of EASE 07, 2007, pp. 83-93.
27 Mendes, E. "The Use of a Bayesian Network for Web Effort Estimation," Proceedings ICWE, Lecture Notes in Computer Science, vol. 4607, pp. 90-104, 2007.
28 Mendes, E., and Mosley, N. "Bayesian Network Models for Web Effort Prediction: a Comparative Study, Transactions on Software Engineering, Vol. 34, Issue: 6, Nov/Dec 2008, pp. 723-737, 2008.
29 Mendes, E., C. Polino, and N. Mosley, "Building an Expert-based Web Effort Estimation Model using Bayesian Networks," 13th International Conference on Evaluation & Assessment in Software Engineering, 2009.
30 Mendes, E., N. Mosley, and S. Counsell, "Investigating Web Size Metrics for Early Web Cost Estimation," Journal of Systems and Software, vol. 77, pp. 157-172, 2005.
31 Mendes, E., Mosley, N. and Counsell, S.: The Need for Web Engineering: An Introduction, Web Engineering, Springer-Verlag, Mendes, E. and Mosley, N. (Eds.) ISBN: 3-540-28196-7, pp. 1-28, 2005.

32 Mendes, E., and S. Counsell. Web Development Effort Estimation using Analogy, Proc. 2000 Australian Software Engineering Conference, 203-212, 2000.
33 Mendes, E., and B.A. Kitchenham. Further Comparison of Cross-company and Within-company Effort Estimation Models for Web Applications, Proc. IEEE Metrics, 348-357, 2004.
34 Mendes, E., and N. Mosley. Does the Linear Size Adjustment to Estimated Effort Improve Web Applications Effort Estimation Accuracy?, Special Issue of the Journal of Computational Methods in Science and Engineering, 5(1), 171-184, 2005.
35 Mendes, E., and N. Mosley. Web Cost Estimation: principles and applications. Web Engineering – Principles and Techniques, Idea Group, Inc., Mehdi Khosrow-Pour and Jan Travers (Eds.), 182-202, 2005.
36 Mendes, E., and N. Mosley. Further Investigation into the Use of CBR and Stepwise Regression to Predict Development Effort for Web Hypermedia Applications, Proc. ACM/IEEE ISESE, Nara, Japan, 79-90, 2002.
37 Mendes, E. N. Mosley "Bayesian Network Models for Web Effort Prediction: A Comparative Study", IEEE Trans. on Soft. Engineering 34 (6), 2008, pp. 723–737.
38 Mendes, E., N. Mosley, S. Counsell, "Investigating Web Size Metrics for Early Web Cost Estimation", Jour. of Systems and Software, 77 (2), 2005, pp. 157-172.
39 Mendes, E., S. Counsell, and N. Mosley. Web Hypermedia Cost Estimation: further assessment and comparison of cost estimation modelling techniques, NRHM, 8, 199-229, 2002.
40 Mendes, E., S. Counsell, and N. Mosley. Towards the Prediction of Development Effort for Hypermedia Applications, Proc. Hypertext 2001, 249 – 258, 2001.
41 Mendes, E., N. Mosley, and S. Counsell. Exploring case-based reasoning for Web hypermedia project cost estimation, IJWET, 2(1), 117-143, 2005.
42 Mendes, E., N. Mosley, and S. Counsell. A Replicated Assessment of the Use of Adaptation Rules to Improve Web Cost Estimation, Proc. ISESE, 100-109, 2003.
43 Mendes, E., N. Mosley, and S. Counsell. Early Web Size Measures and Effort Prediction for Web Costimation, Proceedings of the IEEE Metrics Symposium, 18-29, 2003.
44 Mendes, E., N. Mosley, and S. Counsell. Comparison of Length, complexity and functionality as size measures for predicting Web design and authoring effort, IEE Proc. Software, 149(3), June, 86-92, 2002.
45 Mendes, E., N. Mosley, and S. Counsell. The Application of Case-Based Reasoning to Early Web Project Cost Estimation, Proc. Compsac'02, 393-398, 2002.
46 Mendes, E., N. Mosley, and S. Counsell. Web metrics - Metrics for estimating effort to design and author Web applications. IEEE MultiMedia, January-March, 50-57, 2001.
47 Mendes, E., N. Mosley, and S. Counsell. Using an Engineering Approach to Understanding and Predicting Web authoring and Design. Proc. HICSC, 2001.
48 Mendes, E., N. Mosley, and I. Watson. A Comparison of Case-Based reasoning Approaches to Web Hypermedia Project Cost Estimation, Proc. WWW'02, 2002.
49 Mendes, E., I. Watson, C. Triggs, N. Mosley, and S. Counsell. A Comparative Study of Cost Estimation Models for Web Hypermedia Applications, ESE, 8(2), 163-196, 2003.
50 Montironi, R., W. F. Whimster, Y. Collan, P. W. Hamilton, D. Thompson, and P. H. Bartels, "How to develop and use a Bayesian Belief Network," Journal of clinical pathology, vol. 49, p. 194, 1996. E. Mendes, "A Comparison of Techniques for Web Effort Estimation," in Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on, 2007, pp. 334-343.

51 Murugesan, S., and Deshpande, Y. (eds.), Web Engineering, Managing Diversity and Complexity of Web Application Development, Lecture Notes in Computer Science 2016, Springer Verlag, Heidelberg, Germany, 2001.
52 Pendharkar, P.C., Subramanian, G.H., and Rodger, J.A.: A Probabilistic Model for Predicting Software Development Effort, IEEE Trans. Software Eng. Vol. 31, No. 7, 615-624, 2005.
53 Rajabally, E., P. Sen, S. Whittle, and J. Dalton, "Aids to Bayesian belief network construction," in Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference, 2004, pp. 457-461 Vol.2
54 Reifer, D.J. "Web Development: Estimating Quick-to-Market Software", IEEE Software, Nov.-Dec., 57-64, 2000.
55 Reynolds, P.D. A Primer in Theory Construction, Pearson Education, 2007.
56 Ruhe, M., R. Jeffery, and I. Wieczorek. Cost estimation for Web applications, Proceedings ICSE 2003, 285-294, 2003.
57 Sagrado, J.D., and S. Moral, "Qualitative combination of bayesian networks," International Journal of Intelligent Systems, pp. 237--249, 2003.
58 Studer, R., Benjamins, V.R. and Fensel, D.: Knowledge engineering: principles and methods. Data & Knowledge Engineering, vol. 25, 161-197, 1998.
59 Tang, Z., and B. McCabe: Developing Complete Conditional Probability Tables from Fractional Data for Bayesian Belief Networks, Journal of Computing in Civil Engineering, 21(4), pp. 265-276, 2007.
60 Woodberry, O., A. Nicholson, K. Korb, and C. Pollino, "Parameterising Bayesian Networks," in Australian Conference on Artificial Intelligence, 2004, pp. 1101-1107.