Qualitative Causal Diagrams for Requirements Engineering

Tim Menzies*
Artificial Intelligence Department,School of Computer Science & Engineering, University of NSW, Sydney, Australia, 2052;
Email: timm@cse.unsw.edu.au; Url: www.cse.unsw.edu.au/~timm

June 18, 1997

Abstract

We characterise the software construction process as the negotiation of consistent specifications from a background space of possible conflicting requirements. This characterisation can be operationalised using logical abduction over qualitative causal diagrams.

1 Introduction

Specifications are very different to requirements. While a specification must be complete and consistent, requirements are often not. We characterise the process of requirements engineering (RE) as the extraction of coherent specifications from a space of possibly conflicting requirements.

In the framework proposed here:

- Users agree to a lexicon of terms using repertory grids [10] and a library of desired behaviour.
- Users make statements $\mathcal{S}$ connecting the terms. Each statement is tagged with the name of its authors and is then expanded into a qualitative causal diagram: the space of inferences it condones. That is, a single statement contributes many edged and vertices to a qualitative causal diagram representing the requirements space. Each edge of the qualitative causal diagram is labeled with the unique id of the statements that created it.
- The qualitative causal diagram is under-specified and it may be possible to find contradictions within it. Hence, we next explore the diagram looking for *specifications*; i.e. consistent portions of the qualitative causal diagram that can reproduce some of the desired behaviour. Competing specifications are assessed via their ability to cover the desired behaviour in the cheapest manner.

*3900 words. Submitted to the The Second Australian Workshop on Requirements Engineering (AWRE'97) Monday 27th October 1997 CSIRO-Macquarie University Joint Research Centre for Advanced Systems Engineering (JRCASE) Macquarie University Sydney, Australia.

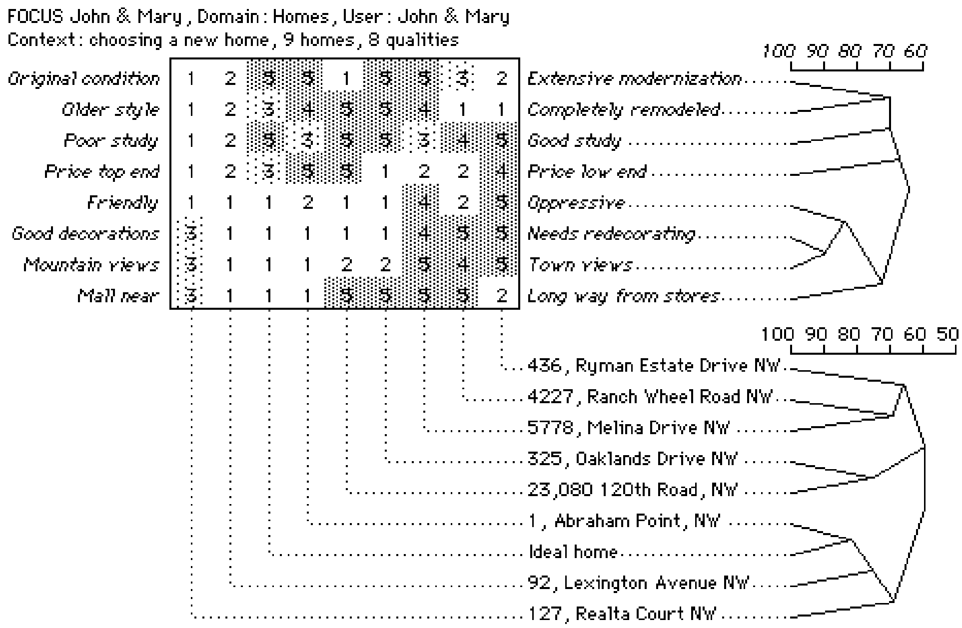

Figure 1: Repertory grids. Generated from the WebGrid WWW server [20].

This article is structured as follows. The use of repertory grids and the acquisition of test suite knowledge is discussed in §2.1 and §2.2 respectively. The base algorithm for processing qualitative influence diagrams is discussed in §3. This technique is applicable to requirements represented in a form which can be translated into a qualitative influence diagram. §4 offers may examples of many such forms.

2 Preliminaries

This section describes the work required before building the qualitative causal diagrams.

2.1 Acquiring the Lexicon with Repertory Grids

Shaw's *repertory grids* can support conflict negotiation of terms in a lexicon of different experts. Experts are asked to identify dimensions along which items from their domain can be distinguished. The two extreme ends of these dimensions are recorded left and right of a grid. New items from the domain are categorised along these dimensions. This may lead to the discovery of new dimensions of comparisons from the expert which, in turn, will cause the grid to grow [10]. For example, based on how an expert scaled some example houses, we can see from the repertory grid of Figure 1 that the `ideal home` is closest to `1, Abraham Point, NW`. Once the dimensions stabilise, and a representative sample of items from the domain have been categorised, then the major distinctions and terminology of a domain have been defined. Inconsistencies are

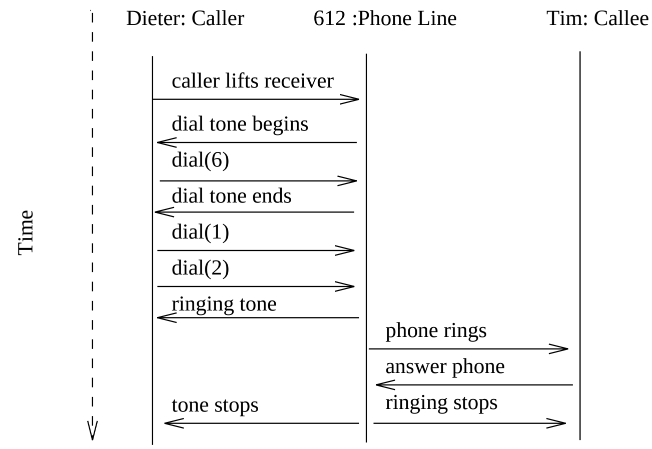

Figure 2: A use case expressed as a sequence diagram.

reported if the categorisations are significantly difference. Using this technique, Gaines & Shaw [10] can detect four classes of inconsistencies in terminology:

- Consensus: same item, same categorisations;
- Correspondence: (a.k.a. synonyms) items with different names, but the same categorisation;
- True conflict: same items, different categorisations;
- Contrast: different items, different categorisations

Shaw (personal communication) reports that resolving terminological disputes is very easy with repertory grids. Once the conflicting repertory grids are shown to the experts, they can quickly (i) see why they differ; and (ii) propose some revision to their terminology to resolve their differences.

2.2 Acquiring the Test Cases

Our approach will assess conflict spaces via their coverage of known or desired behaviour. This section describes how we might acquire such a library.

Firstly, for RE of physical systems (e.g. blast furnaces, computer chips from third-party manufacturers, medical diagnosis systems), such test cases libraries might be naturally available.

A second class of test cases may come from government legislation. That is, in certain circumstances, certain events must happen.

A third class of test cases may come from scenario-driven analysis methods such as uses cases [2, 14, 18, 19]. A use case has a very simple text structure. Developers write a short "story" (say less than 2 pages) describing some flow of events within their system. The text of the use case is mapped into classes via *sequence diagram* such as Figure 2. The arrows on a sequence diagram represent the flow of events of the use case text. Each such arrow implies a method at the sender end and a method at the receiver end. Note the specific data contained

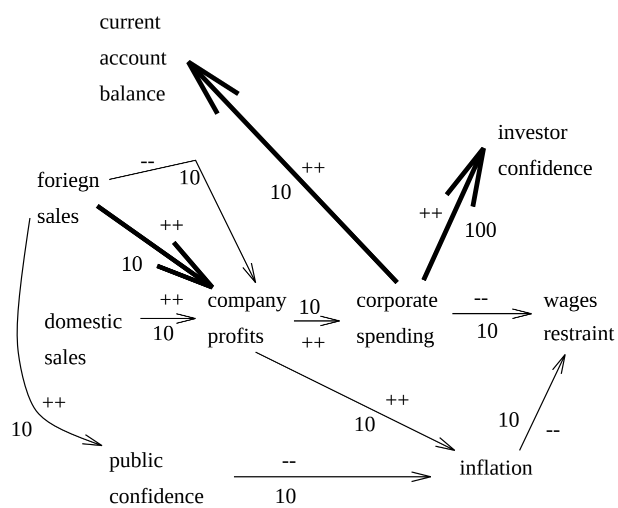

Figure 3: Some economics knowledge.

in Figure 2 (dialing of `612`, specific data values like `Dieter` and `Tim`). Use cases can also be used to extract test cases from business users.

3 Recognises and Negotiating Conflicts

This section describes the use of HT4 abductive inference engine to generate the specifications from the requirements represented as qualitative causal diagrams.

3.1 Building the Conflict Space

HT4 searches for consistent portions of some background theory which are relevant to some task. Abduction is the search for assumptions $\mathcal{A}$ which, when combined with some theory $\mathcal{T}$ achieves some set of goals $\mathcal{OUT}$ without causing some contradiction [7]. That is:

- EQ_1: $\mathcal{T} \cup \mathcal{A} \vdash \mathcal{OUT}$;
- EQ_2: $\mathcal{T} \cup \mathcal{A} \not\vdash \bot$.

HT4 caches the proof trees used to satisfy EQ_1 and EQ_2. These are then sorted into *worlds* $\mathcal{W}$: maximal consistent subsets (maximal with respect to size). Each world condones a set of inferences. A world's *cover* is the size of the overlap between $\mathcal{OUT}$ and that world. In the case of multiple worlds being generated, a customisable assessment operator is used to select the preferred world(s).

For example, consider the task of achieving certain $\mathcal{OUT}$puts using some $\mathcal{IN}$puts across the knowledge shown in Figure 3. In that figure:

- $\mathtt{x} \overset{++}{\rightarrow} \mathtt{y}$ denotes that `y` being `up` or `down` can be explained by `x` being `up` or `down` respectively;

$\mathcal{P}[1]$: `domesticSalesDown, inflationDown`
$\mathcal{P}[2]$: `foriegnSalesUp, publicConfidenceUp, inflationDown`
$\mathcal{P}[3]$: `domesticSalesDown, companyProfitsDown, corporateSpendingDown, wagesRestraintUp`
$\mathcal{P}[4]$: `domesticSalesDown, inflationDown, wagesRestraintUp`
$\mathcal{P}[5]$: `foriegnSalesUp, publicConfidenceUp, inflationDown, wagesRestraintUp`
$\mathcal{P}[6]$: `foriegnSalesUp, companyProfitsUp, corporateSpendingUp, investorConfidenceUp`

Figure 4: Proofs from Figure 3 connecting $\mathcal{OUT}$= {`investorConfidenceUp`, `wagesRestraintUp`, `inflationDown`} back to $\mathcal{IN}$puts= {`foriegnSalesUp`, `domesticSalesDown`}.

- `x` $\overset{--}{\rightarrow}$ `y` denotes that `y` being `up` or `down` could be explained by `x` being `down` or `up` respectively.

Each edge in Figure 3 is augmented with two pieces of meta-information which are explored subsequently. However, the notation is introduced now:

1. Each edge is annotated with a heuristic weight representing how expensive it is to make that inference. Most edges have cost 10, but the edge `corporateSpending` $\overset{++}{\rightarrow}$ `investorConfidence` requires a large amount of book-keeping by associate accountancy packages to measure the subjective measure `investorConfidence`. Hence, this edge has a weight of 100[1]. We will discuss the use of this edge weight later.

2. Figure 3 is a combination of the opinions of two authors: *Dr. Thick* (whose contribution is drawn with thick lines) and *Dr. Thin* (whose contribution is drawn with thin lines). Observe the apparent conflict in the middle of Figure 3 on the left-hand-side. *Dr. Thick* believes:

   `foriegnSales` $\overset{++}{\rightarrow}$ `companyProfits`

   while *Dr. Thin* believes

   `foriegnSales` $\overset{--}{\rightarrow}$ `companyProfits`

In the case of the observed $\mathcal{OUT}$puts being:

{`investorConfidenceUp, wagesRestraintUp, inflationDown`},

and the observed $\mathcal{IN}$puts being:

{`foriegnSalesUp, domesticSalesDown`}.

HT4 can connect $\mathcal{OUT}$puts back to $\mathcal{IN}$puts using the proofs of Figure 4. These proofs may contain controversial assumptions; i.e. if we can't believe that a variable can go `up` and `down` simultaneously, then we can declare the known values for `companyProfits` and `corporateSpending` to be controversial. Since `corporateSpending` is fully dependent on `companyProfits` (see Figure 3), the key conflicting assumptions are {`companyProfitsUp, companyProfitsDown`} (denoted *base controversial assumptions* or $\mathcal{A}_b$). We can used $\mathcal{A}_b$ to find consistent belief sets called worlds $\mathcal{W}$ using an approach inspired by the ATMS [4]. A

[1]It is easier to measure `publicConfidence` via simple telephone surveys. Hence, the cost of its input-edge is only 10.

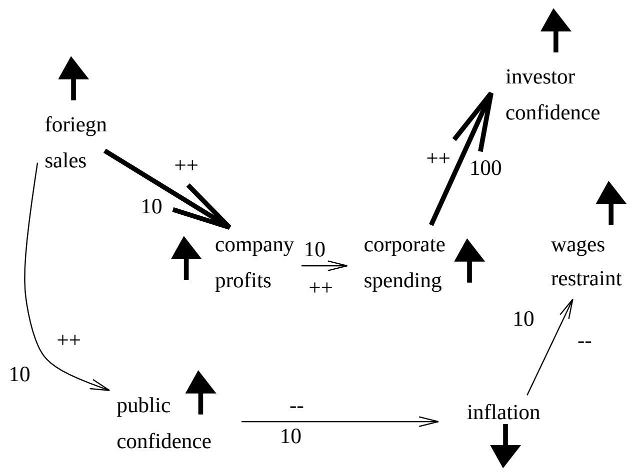

Figure 5: World #1 is generated from Figure 3 by combining $\mathcal{P}[2]$, $\mathcal{P}[5]$, and $\mathcal{P}[6]$. World #1 assumes `companyProfitsUp` and covers 100% of the known $\mathcal{OUT}$puts.

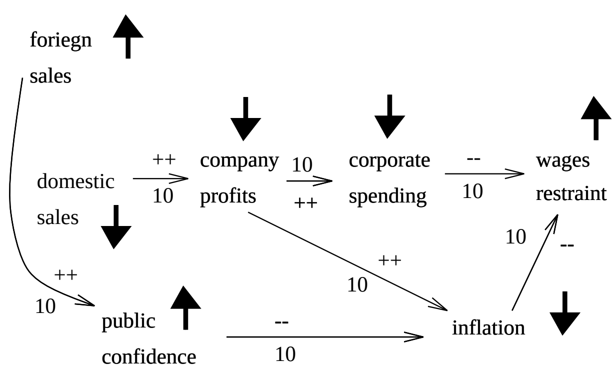

Figure 6: World #2 is generated from Figure 3 by combining $\mathcal{P}[1]$, $\mathcal{P}[2]$, $\mathcal{P}[3]$, and $\mathcal{P}[4]$. World #2 assumes `companyProfitsDown` and covers 67% of the known $\mathcal{OUT}$puts.

proof $\mathcal{P}[i]$ is in $\mathcal{W}[j]$ if that proof does not conflict with the environment $\mathcal{ENV}[j]$. In our example, $\mathcal{ENV}[1]$={`companyProfitsUp`} and $\mathcal{ENV}[2]$={`companyProfitsDown`}. Hence, $\mathcal{W}[1]=\{\mathcal{P}[2], \mathcal{P}[5], \mathcal{P}[6]\}$ and $\mathcal{W}[2]=\{\mathcal{P}[1]\ \mathcal{P}[2]\ \mathcal{P}[3], \mathcal{P}[4]\}$ (see Figure 5 and Figure 6). Note that while the background theory (Figure 3) may be inconsistent, the generated worlds are guaranteed to be consistent.

3.2 Negotiating Conflicts

Once the worlds are generated, they must be assessed. The worlds of Figure 5 and Figure 6 tell us:

- *Dr. Thin*'s contributions can be found in two worlds; i.e. with respect to the problem of $\mathcal{OUT}$puts= {`investorConfidenceUp, wagesRestraintUp,` `inflationDown`}, and $\mathcal{IN}$puts= {`foriegnSalesUp, domesticSalesDown`}, a single author's opinions are inconsistent.

- Both authors contributions exist in the same consistent world ($\mathcal{W}[1]$); i.e. the apparent conflict of *Dr. Thick* and *Dr. Thin* did not matter for the analysed problem. If this was true for all the analysed problems, then we could declare that for all practical purposed, *Dr. Thin* and *Dr. Thick* are not really disagreeing.

- *Dr. Thin* may wish to review their opinion that

$$\texttt{foriegnSales} \stackrel{--}{\rightarrow} \texttt{companyProfits}$$

since, in terms of the studied problem, this proved to explain less of the required behaviour that *Dr. Thick*'s option that

$$\texttt{foriegnSales} \stackrel{++}{\rightarrow} \texttt{companyProfits}.$$

On the other hand, *Dr. Thin*'s views may be more economical to implement than *Dr. Thick*'s. Note how $\mathcal{W}[1]$ uses the

$$\texttt{corporateSpending} \stackrel{++}{\rightarrow} \texttt{investorConfidence}.$$

edge while $\mathcal{W}[2]$ ignores it. This edge is very expensive to use (weight=100). Hence, we could accept *Dr. Thin*'s view since we may elect to trade-off the completeness of *Dr.Thick*'s views for the economy of *Dr. Thin*'s views.

3.3 Advantages

HT4 has technical advantages over other conflict resolution approaches:

- Easterbrook [6] lets users enter their requirements into an explicitly labeled viewpoints. He makes the simplifying assumption that all such viewpoints are internally consistent. HT4 has no need for this, potentially, overly-restrictive assumption. HT4 can handle inconsistencies within the opinions of a single user. That is, HT4 can analyse conflicts at a finer granularity than approaches based on manually-entered viewpoints (e.g Easterbrook or Finkelstein *et. al.* [8]).

- Easterbrook's Synoptic tool only permits comparisons of two viewpoints [6, p113]. HT4 can compare N viewpoints.

- We have found that it easier to build efficient implementations [15, 16] using the above graph-theoretic approach that using purely logical approaches (e.g. [13]).

- We will argue below (§4) that HT4 is widely applicable to many representations.

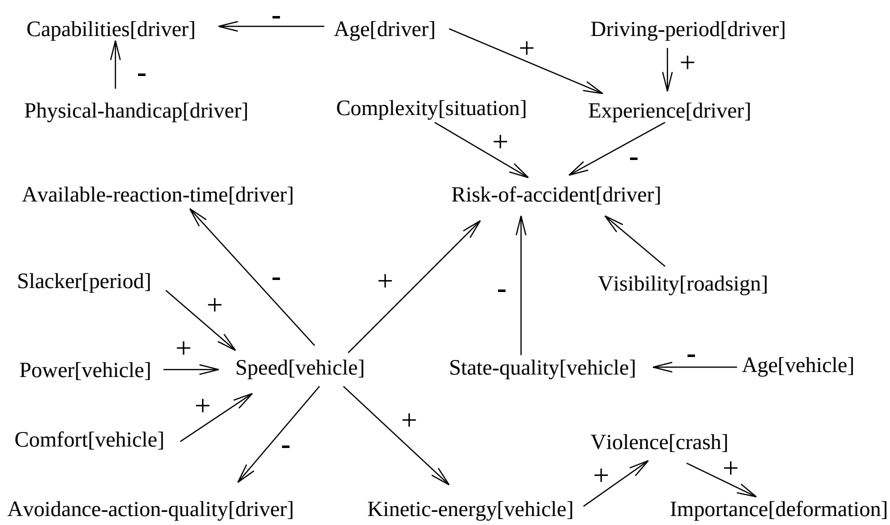

Figure 7: A graphical representation of topoi using the 3DKAT. From [5].

4 Generality

HT4 places few restrictions on the representations it can process. The above process is defined for any representation that can be mapped into a qualitative casual diagram such as certain constructs from natural lanugage (§4.1); rule-based systems (§4.2); the goal graphs of non-functional requirements (§4.3); Harel statecharts (§4.4); and a simple variant on entity-relationship diagrams (§4.5).

4.1 Natural Language

Topoi graphs are a visual representation of a common natural language construct describing gradual knowledge; i.e. statements of the form: (i) the more X, the more Y; (ii) the less X the less Y; (iii) the more X, the less Y; or (iv) the less X the less Y. Dieng *et. al.*call such statements *topoi* and give numerous examples from their records of interviews with experts [5]. For example:

- The more there is water infiltration in the roadway body, the worse the foundation risks to be.
- The higher the speed of the vehicles, the more important the measure of importance relative to the roadway comfort.
- When the geometry increases, the mass increases and the frequency decreases.
- If there is a punctual undressing and if the roadway is between five and fifteen years old, then the causes "too old coating" is all the more certain since the roadway is older.

Dieng *et. al.* describe 3DKAT, a knowledge acquisition tool for the graphical presentation of topoi. A sample 3DKAT topoi is shown in Figure 7. Clearly, a visual presentation of topoi can be processed in the manner of §3.

```
if     infant or moron
then   not legally_responsible.

if      age > 7
then    infant.

if      legally_responsible and guilty
then    jail.

if      motive and means and
        opportunity and witnesses
then    guilty.

if      guilty
then    jail.

if      guilty and
        not legally_responsible
then    not jail.
```

Figure 8: A propositional system.

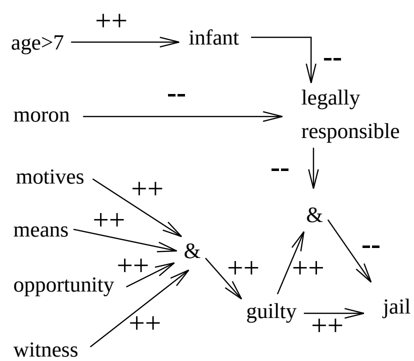

Figure 9: A qualitative causal diagram which can be inferred from Figure 8. Note that this diagram is compatible with Figure 3.

4.2 Rules

Dieng *et. al.* claim that topoi are a valid description for a range of representations such as production rules.

A diagram showing the dependencies between literals in a propositional knowledge base is a qualitative causal digram. The qualitative causal diagram of the propositional system of Figure 8 is shown in Figure 9. In this diagram:

- $\mathtt{X} \stackrel{++}{\rightarrow} \mathtt{Y}$ denotes that an increase in the level of belief in `Y` can be explained via an increase in the level of belief in `X`.

- $\mathtt{X} \stackrel{--}{\rightarrow} \mathtt{Y}$ denotes that an decrease in the level of belief in `Y` can be explained via an increase in the level of belief in `X`.

Diagrams such as Figure 9 are often used to understand the logical structure of a rule base [11] or for optimising the inferencing process [9]. Dieng *et. al.* [5]

```
''Monitoring and control''
Preconditions:
        ''Monitoring instrumentation''
        ''Control limits''
        ''Algorithms''

Postconditions:
        ''If the function is stable, checks the performance and reports
        it, otherwise stabilises the function by controlling the
        configuration or environment. May also report predicted future
        undesirable states.''

Effects on quality attributes:
        Assurance     : pros ''Avoids undesirable states''
        Performance   : cons ''Needs additional processing in short term''
                        pros ''Improves performance in long term via tuning''
        Timeliness,
        Affordability : cons ''More effort to specify''
                        cons ''More effort to develop''
                        cons ''More effort to verify''
```

Figure 10: NFR quality knowledge: strategy knowledge from QARCC. From [1].

argue that many rule bases can be converted to topoi without significant loss of functionality.

Topoi graphs are often indeterminate. As a result, they can generate incompatible conclusions. For example in the case of `guilty` and `moron` then Figure 9 can infer both an increase and a decrease in our level of belief in `jail`. Inconsistent theories need to manage their inconsistent assumptions in separate sets; e.g. the *worlds generation process* of §3.

4.3 Goal Graphs for Non-Functional Requirements

Functional requirements can be measured via executing and measuring a program. Non-functional requirements (NFR) such as portability, evolvability, development affordability, security, privacy, or reusability cannot be assessed with respect to the current version of the working program. For example, consider the NFR of maintainability. Maintainability can only be definitively assessed in retrospect; i.e. only after delivery has occoured and we have some track record of the system's performance in the field. Nevertheless, during initial construction, we may still want to assure ourselves as to the potential maintainability of the system.

Two example of NFR systems are Chung & Nixon's *goal graph* [3] approach and Boehm's *et. al.* QARCC tool [1]:

- The QARCC KB represents the concerns of different *stakeholders* (e.g. user, customer, developer, developer, maintainer, interfacer, and general public) and the quality attributes which map into those concerns. For example, maintainers are mostly concerned with evolvability and portability while customers and developers are mostly concerned with development affordability and reusability. A *strategy* fragment of a QARCC quality knowledge base is shown in Figure 10. These strategy fragments are mapped into different stakeholders (e.g. developers and customers

A. A depth-less chart. *B. Chart with depth.*

Figure 11: Harel statecharts. The initial state is modeled as a filled-in circle. Adapted from [12].

both worry about `Affordability`). These strategies are then explored looking for conflicts such as `Performance` vs `Affordability` trade-offs.

- Goal graphs contain similar trade off information to the QARCC strategy fragments, but do not explicitly model stakeholders.

Like our topoi graphs, NFRs are indeterminate. For example, returning to Figure 10, consider a trade-off between competing `pros` and `cons`. Without exact information regarding the relative sizes of the `pro` or `con` influence, we should fork multiple worlds and assess them in the manner of §3. Note that such an assessment procedure is hard-wired within the QARCC and *goal graph* systems.

4.4 Using StateCharts

Harel's statecharts [12] are an extension of a state transition diagram to a graph with *triggers*, *guards*, *depth, orthogonality* and *broadcast communication*:

- Edges from a state are augmented with a *trigger*. If such a trigger event occurs whilst the system is in a state, then that edge is traversed. For example, while in state `A` of Figure 11.A, the `f` event will take us to state `B`.

- A statechart edge can have conditionals that *guard* each edge (denoted `[condition]`). For example, while in state `A` of Figure 11.A, the `g` will only take us to state `C` if the guard `x` is true.

- If an edge is traversed, a procedure can be called as a side-effect (called a *broadcast*). Non-binary relationships can be represented by broadcast edges since they may connect sets to sets rather than just individuals to individuals.

- A state in a statechart can contain nested states. The nested states is said to be at a greater *depth* than the outer state. Using depth, we can simplify (e.g.) Figure 11.A to Figure 11.B. Nested states are read the same as inclusion in Venn diagrams. Inferences made about outer-nested states apply to the inner-nested states they contain.

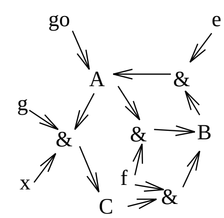

Figure 12: Figure 11 modeled in a manner compatible with Figure 3.

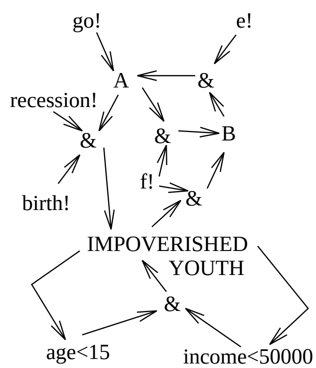

Figure 13: An extension to Figure 12. Events are denoted with an exclamation mark and states are in UPPER CASE. Other vertices are variable settings.

- States can be divided into *orthogonal* components. If such a divided state is entered, then the system is said to be in *all* the divided components. This is a useful tool for reducing the size of some charts. It also is a natural tool for modeling parallelism.

Figure 12 shows us that depth, guards, and triggers of the Harel statechart of Figure 11.B can be naturally modeled in a diagram compatible with Figure 3 (leaving broadcasts and orthogonality as open research issues). In Figure 12, triggers are modeled as conjunctions and guards are modeled as conjunctions on the conjunctions. A special state `go` is used to denote the initial state. Nested states are unwound to create statecharts of depth 0 (so the translation process would be from Figure 11.B to Figure 11.A to Figure 12).

Figure 13 shows a possible variant on Figure 12 in which being in a state `X` implies certain settings. For example, the `ImpoverishedYouth` state could imply `age<15` and `income<5000`. If we enter this state, then these variables should be set. Further, we can infer what state we should be in from an analysis of the settings that imply a state.

4.5 Entity-Relationship Diagrams

In terms of states in one entity effecting states in other entities, standard ER diagrams such as Figure 14 are poor candidates for this technique. For example:

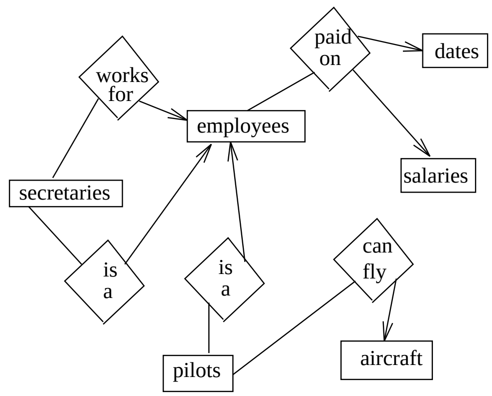

Figure 14: An entity-relationship diagram.

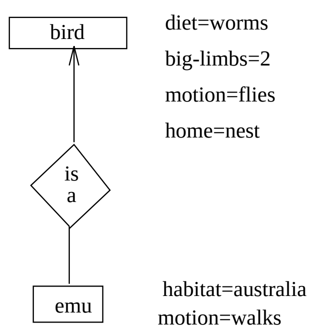

Figure 15: An ER diagram with defaults for attributes.

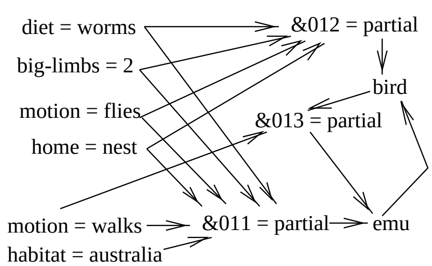

Figure 16: Generated from Figure 15.

- Is there any concept of directionality?; e.g. is Figure 14 saying that `pilots` can effect `aircraft` but not visa versa?. Such directionality effects are not defined in standard ER.
- Can any state in one side of a relationship effect any state on the other side of the relationship? If so, then ER diagrams threaten to be to permissive; i.e. offer explanations for any state change.

However, with some knowledge of how they are processed, variants of ER are suitable for this approach. For example, suppose we know (i) default values for all the attributes in a relationship; and (ii) that these defaults rarely change. Then we can use the extended ER diagram as a classification tool in a manner similar to Minsky's frame proposal [17].

For example, the qualitative causal diagram of Figure 15 could be converted into the intermediary diagram of Figure 16. As with the qualitative reasoning domain, extra edges have to be added to the intermediary to manage representation-specific semantics; e.g. if `emu` then `bird` via the `isa` link. Also, note the *partial conjunction* vertices in Figure 16. Partial conjunctions can be satisfied by *any* of their pre-conditions (whereas standard conjunctions need all their pre-conditions satisfied). We could infer to `emu` if told `habitat=australia` but we did not know anything yet about `motion`. However, a downstream assessment routine may elect to favour the conclusions with the most pre-conditions of the partial conjunctions satisfied.

5 Conclusion

We characterise the processing of RE as the construction and assessment of alternative specifications (a.k.a. HT4 worlds) from a requirements space represented as a qualitative causal model. We argue that this technique is very general since it can many common notations from software engineering can be represented as qualitative causal diagrams. One nice feature of our approach is that the same architecture can be used for processing functional and non-functional requirements.

References

[1] B. Boehm. Aims for Indentifying Conflicts Among Quality Requirements. In *IEEE Software*, March 1996.

[2] G. Booch. *Object Solutions: Managing the Object-Oriented Project.* Addison-Wesley, 1996.

[3] L. Chung and B.A. Nixon. Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach. In *Proceedings of ICSE '95: the International Conference on Software Engineering*, pages 25–36, 1995.

[4] J. DeKleer. An Assumption-Based TMS. *Artificial Intelligence*, 28:163–196, 1986.

[5] R. Dieng, O. Corby, and S. Lapalut. Acquisition and Exploitation of Gradual Knowledge. *International Journal of Human-Computer Studies*, 42:465–499, 1995.

[6] S. Easterbrook. *Elicitation of Requirements from Multiple Perspectives.* PhD thesis, Imperial College of Science Technology and Medicine, University of London, 1991. Available from *http://research.ivv.nasa.gov/ ∼steve/papers/index.html.*

[7] K. Eshghi. A Tractable Class of Abductive Problems. In *IJCAI '93*, volume 1, pages 3–8, 1993.

[8] A. Finkelstein, D. Gabbay, A. Hunter, J. Kramer, and B. Nuseibe. Inconsistency Handling In Multi-Perspective Specification. *IEEE Transactions on Software Engineering*, 20(8):569–578, 1994.

[9] C.L. Forgy. RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*, pages 17–37, 19 1982.

[10] B.R. Gaines and M.L.G Shaw. Comparing the Conceptual Systems of Experts. In *IJCAI '89*, pages 633–638, 1989.

[11] C. Grossner, A. Preece, P. Gokul Chander, T. Radhakrishnan, and C. Suen. Exploring the Structure of Rule Based Systems. In *Proc. 11th National Conference on Artificial Intelligence (AAAI-93)*, pages 704–709. MIT Press, 1993.

[12] D. Harel. On Visual Formalisms. In J. Glasgow and B. Chandrasekaran N.H. Narayanan, editors, *Diagrammatic Reasoning*, pages 235–271. The AAAI Press, 1995.

[13] A. Hunter and B. Nuseibeh. Analysing Inconsistent Specifications. In *International Symposium on Requirements Engineering*, pages 78–86, 1997.

[14] I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley, 1992.

[15] T.J. Menzies. On the Practicality of Abductive Validation. In *ECAI '96*, 1996.

[16] T.J. Menzies. Applications of Abduction: Knowledge Level Modeling. *International Journal of Human Computer Studies*, 45:305–355, September, 1996.

[17] M. Minsky. A Framework for Representing Knowledge. In *The Psychology of Computer Vision*, 1975.

[18] K.S. Rubin and A. Goldberg. Object Behavior Analysis. *Communications of the ACM*, 35(9), 1992.

[19] J. Rumbaugh. Getting Started: Using Use Cases to Capture Requirements. *JOOP*, pages 8–23, 1994.

[20] M. Shaw. *WebGrid: a WWW PCP Server.* Knowledge Systems Institute, University of Calgary, `http://Tiger.cpsc.ucalgary.ca/WebGrid/WebGrid.html`.

Some of the Menzies papers can be found at *http:// www.cse.unsw.edu.au/ ~timm/pub/ docs/papersonly.html.*