ASSUMPTIONS IN DESIGN AND IN DESIGN RATIONALE

DAVID C. BROWN
Computer Science Department
WPI, Worcester, MA 01609, USA
dcb@cs.wpi.edu

1. Definition

What is an assumption? From dictionaries we get something like:

- an assuming that something is true;
- a fact or statement (as a proposition, axiom, postulate, or notion) taken for granted.

From a more formal AI point of view “an assumption is something which is accepted in the absence of evidence to the contrary” [Ramsey 1988].

2. Motivation

Often assumptions are used as a starting point for a course of action or reasoning. The reliability of the result depends on the reliability of the assumptions.

Design decisions are usually influenced by a lot of information. Some of it is given in the problem. Some of it comes from the best practices of the field, often imposed as design standards. But some information comes from the experiences of the designers and implementers. That experience suggests configurations to prefer in different situations, and familiar components to use.

Experiences lead to preferences being formed. With those preferences come assumptions. Designers tend to assume normal situations. They tend to make assumptions about the match between the current design situation and one where their chosen technique worked well before. They tend to make incorrect abstractions across all the situations where particular techniques worked well before. Abstractions, by definition, lose detail. This can be done by assuming that some key detail isn't relevant.

Such assumptions are often not deliberate, but a form of tacit knowledge [Barbiero 2004] underlying expert skill, or are rarely stated beliefs about things.

Design reuse can violate assumptions, as conditions that were true, or were assumed to be true originally may no longer be the case in the new design context [Charlton & Wallace 2000]. This may cause the component to no longer function as expected, for example. Changes to designs can also violate assumptions.

Hence, in order to support design reuse, or changes to designs, there's a need to make assumptions explicit and visible [Burge & Brown 2006]. If one could record assumptions, then they could be attached to design decisions. That is easier said than done, as many of the assumptions are made without knowing that they've been made. Even explicitly made assumptions can be made as a matter of course, as part of a habitual design process.

We need to find ways to retrieve or infer and then collect whatever assumptions are being made. The user needs to be warned about their (possible) assumptions at design time, and records of the acknowledged assumptions need to be attached firmly to the design: perhaps the suspected assumptions too.

3. Causes

Why are assumptions made? Here is a list of most of the major reasons, along with sample examples:

- lack of knowledge
  - e.g., not knowing that a particular motor runs hot might allow an assumption that thermal expansion of a component is not an issue.
- to simplify the problem and constrain the design space
  - e.g., assume friction is negligible
  - e.g., assume the problem is decomposable
  - e.g., assume right-handed users
- to standardize the problem
  - e.g., with “standard” Non-Functional Requirements
  - e.g., with requirements borrowed from similar past projects
- to make a general statement rather than a specific one
  - e.g., assuming that differences are not significant
- different tools inherit/encourage different assumptions
  - e.g., sketching vs. CAD vs. flowcharting
  - e.g., models may assume no mass or no friction [Addanki et al. 1989]
- cultural pressure

  - e.g., an assumption that certain qualities are preferred, or even required, due to design trends/fads, such as “streamlining”
- the arrogance of experts
  - e.g., the assumption that they aren't making assumptions!
  - e.g., familiarity, old technology, and continued successful deployment leads to assumptions about a design [Petroski 2006]
- ambiguity in Requirements
  - e.g., an assumption about what the requirements mean
- assumptions are usually relative to a situation
  - e.g., classifying the design context can lead to assumptions, and misclassifying it can lead to incorrect assumptions.
- assumptions come from rules, norms and conventions
  - e.g., rules come with a set of assumptions about applicability, and so rule use adopts those assumptions
- assumptions come from expectations
  - e.g., expected properties/performance of a design may induce assumptions about requirements
- the desire to break away from routineness
  - e.g., deliberately made, perhaps incorrect, assumptions might take the design into a different search space, leading to creative results.
- assumptions are the norm in everyday activity
  - e.g., life would be too complex if we didn't make them, so we are conditioned to do so. “We are automatic assumption machines.” [Niquette 1996]

4. Consequences

Assumptions during designing about the “mode of deployment” [Chandrasekaran & Josephson 2000] of a device will lead to one that functions incorrectly, while such assumptions during reuse will lead to the device not behaving as expected in the new context. The same sort of argument applies to design processes. Note that different stakeholders in a design process, such as those involved in a Concurrent Engineering team, will probably have different assumptions, leading to more problems.

5. Detection & Capture

How to detect and capture assumptions are certainly ongoing research challenges, but detection methods might include:

- Noticing mismatches between actual and intended behaviours;
- Challenging assumptions by using “what if” questions;
- Inference using design rationale.

and capture methods might be:

- direct (explicit capture)
- by inference (implicit capture)
  - e.g., infer from design rationale, or other stored knowledge

6. Rationale for Decisions & Assumptions

However they are gathered it's clear that assumptions must be included in Design Rationale [Burge & Brown 2006], as explicit assumptions can be part of the reason for a design decision. The source of assumptions should be included, as, if they are provided explicitly and are found to be incorrect, then the action that should be taken is different from the situation where the assumptions are inferred and found to be incorrect.

In addition, the rationale for an explicit assumption should be kept along with the other rationale. Effectively this is a record of the conditions that triggered the production of the assumption.

7. Conclusion

Petroski points out that "...too great a reliance on successful precedents can lead to failure. Success is not simply the absence of failure; it also masks potential modes of failure." [2006, p.3]. As much design is case-based or close to it, this observation is important: especially as past designs come bundled with the assumptions that allowed them to work. Lehman [2003] states that "Invalid assumptions constitute the primary source of project and system misbehavior or, in the extreme, failure". Consequently, it's clear that assumptions must be captured, tracked and utilized in order to attempt to avoid design problems.

Acknowledgments

Thanks to Janet Burge and Rob Bracewell for encouraging me to write this short submission to this workshop.

References

S. Addanki, R. Cremonini & J.S. Penberthy (1989) "Reasoning about Assumptions in Graphs of Models", *Proc. Int. Jnt. Conf. on AI, IJCAI-89*, pp. 1432-1438.

D. Barbiero (2004) "Tacit knowledge", In: *Dictionary of Philosophy of Mind*, C. Eliasmith (Ed.), <philosophy.uwaterloo.ca/MindDict/tacitknowledge.html>.

J.E. Burge & D.C. Brown (2006) "Rationale-Based Support for Software Maintenance", In: *Rationale Management in Software Engineering*, (Eds.) A.H. Dutoit, R. McCall, I. Mistrik, B. Paech, Springer, pp. 273-296.

C. Charlton & K. Wallace (2000) "Reminding and Context in Design", In: *Artificial Intelligence in Design 2000*, J.S. Gero (Ed.), pp. 596-588.

B. Chandrasekaran & John R. Josephson (2000) "Function in Device Representation", *Engineering with Computers*, Special Issue on Computer Aided Engineering, Vol. 16, pp. 162-177.

M.M. Lehman (2003) "Software Evolution: Cause or Effect", Stevens Memorial Lecture, *ICSM 2003*, Amsterdam, September 2003, <www.cs.mdx.ac.uk/staffpages/mml/711c-StevensL_.ppt>.

P. Niquette (1996) "Discovering Assumptions", <www.niquette.com/books/sophmag/heurist.htm>.

H. Petroski (2006) *Success through Failure*, Princeton University Press, pp. 174-175.

A. Ramsey (1988) *Formal Methods in Artificial Intelligence*, Cambridge University Press.