

Cognitive Dimensions - a short tutorial

PPIG 2008, Lancaster University

Luke Church (Cambridge University) & **Thomas Green** (University of Leeds)

luke@church.name : stuff@greenery.me.uk

Why?

It's easy to look at a piece of technology and say it's got lots of features. Deciding whether it will be usable is harder. We could get a bunch of volunteers and try it - but think of the problems: getting the volunteers is hard enough, but even when you've done that, all you'll find out will be the worst problems. Even worse, you'll have to actually build the technology before you can test it.

So people have developed a number of predictive approaches. They vary greatly but they all try to predict whether technology will be usable, preferably without needing real volunteers and if possible without needing to build the real technology.

Some approaches focus on one type of technology. That's not enough; we need an approach that works for all types of technology, including non-interactive systems (notations) as well as the more high-profile interactive systems.

You're still not out of the wood. So you discover that your technology is problematic. Is it problematic in the same way as some other piece of technology? Will the same solution be relevant to both of them? We need a standardised vocabulary, a few words describing familiar good/bad points.

OK, nearly out of the wood, the trees are thinning out, but we're still not quite there When you've got a vocabulary, what of it? Will a potential usability problem be relevant to the particular use you envisage? We need a standardised, high-level account of what technology can be used for, and we need to relate our vocabulary of good/bad points to the potential uses.

The Cognitive Dimensions framework does that, and is still unique in doing so, we believe.

What ... in 15 seconds

The CDs framework provides

- a standardised framework of 'cognitive dimensions', each relating to usability experience,
- where each dimension is (more or less) independent of each of the others,
- applying to every kind of information artefact (i.e. anything used for storing or manipulating or accessing information - that's most non-natural things);
- a background in the experimental literature of cognitive psychology relating to each of the dimensions;
- a high-level classification of the activities people get up to with technology;
- and an attempt to say, for each activity, which cognitive dimensions are critical to making that activity successful.

The last point is crucial. No feature is good or bad in itself. Until you know what it's to be used for, you can't say whether a device is useably designed or not.

One other point. The dimensions are not brand new discoveries. On the contrary, they will all sound very familiar from your own experience - but you probably haven't named them before, let alone built them into a system. (If you *have*, let's hear about it, please!)

What ... in 2 or 3 hours?!

Teaching cognitive dimensions has a couple of problems. One is that there are too many dimensions to do them all at one go. Perhaps this isn't surprising: after all, using information artefacts is a pretty complex business, so there are lots of aspects to consider. Our solution will be to consider a very restricted subset of dimensions, the ones that are most likely to impinge nastily on what we imagine our audience will be.

The other is that it's very high-level, and doesn't make much sense on its own until you come to try it out. So we intend to spend a good proportion of the time talking about audience examples, preferably from stuff you're building or working with; you give a (succinct!) description of what your stuff is and what it's for, and we - or better still, the rest of the audience - attempt to say something about it. If that turns out to have been useful, you can show your appreciation by stuffing our pockets with gold.

The framework

Here we go then.

Things

Information artefacts include programming languages, spreadsheets, digital libraries, paper-based libraries, filing systems, menu systems, recipe books, clocks and watches, maps,

textbooks, codes and ciphers, ticket machines, diaries, control systems for videos, ovens, microwaves and nuclear power plants, train signalling systems, timetables, telephones, and abstracts for tutorials. And more.

Activities

We recognise 7 activities (used to be 6 but we added comparison)

- searching (as in a database)
- incrementing (adding more data of the same sort – another entry to a database, another line to a program)
- modifying structure (changing the structure of a database, refactoring a program)
- transcribing (copying information from structure to another – coding a flowchart, building a spreadsheet to compute from mathematical formulas)
- exploratory design (building where you're feeling your way)
- exploratory comprehension (trying to work out what something's for, as opposed to the details of how it does it)
- comparison (is this thing like that one?)

Dimensions

Here are brief definitions of most of the cognitive dimensions. They (or some of them) will be explained and illustrated during the tutorial - this list is meant for a later reminder.

Abstraction	types and availability of abstraction mechanisms
Hidden dependencies	important links between entities are not visible
Premature commitment	constraints on the order of doing things
Secondary notation	extra information in means other than formal syntax
Viscosity	resistance to change
Visibility	ability to view components easily
Closeness of mapping	closeness of representation to domain
Consistency	similar semantics are expressed in similar syntactic forms
Diffuseness	verbosity of language
Error-proneness	notation invites mistakes
Hard mental operations	high demand on cognitive resources
Progressive evaluation	work-to-date can be checked at any time
Provisionality	degree of commitment to actions or marks
Role-expressiveness	the purpose of a component is readily inferred

Unpacking the Dimensions

Each of these dimensions turns out to be messy when looked at closely. However, this is one of the advantages of the framework: it can be used at multiple levels of detail. When you want the broadest analysis, it can be summarized in a diagram, as below. When you want more detail the research literature discusses the dimensions and activities in depth.

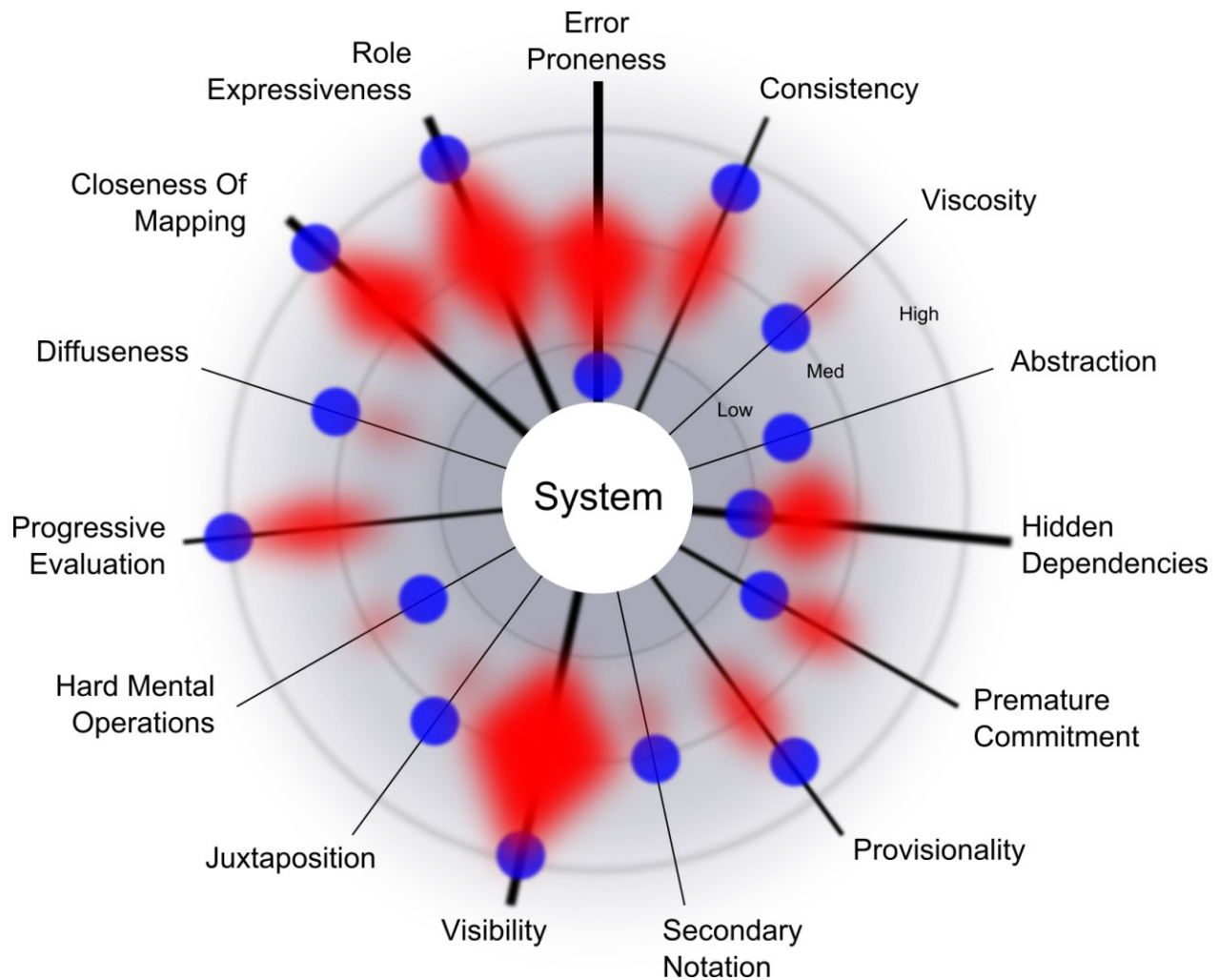


Figure 1 – A diagrammatic summary form of the Cognitive Dimensions. A different diagram is used for each combination of system and activity. The blue dots represent an 'ideal value' for a dimension and the thickness of the lines represents the importance of a dimension for the given activity; this much will be the same for all systems. The red represents the difference between the ideal and the actual value, in the system being analysed, and its width correlates to the importance. The intuition is that the overall amount of red correlates to how bad an interface is, and how important that badness is for a specific dimension.

For this tutorial, we'll stick to a moderate amount of detail. To give a flavour as to how the dimensions can become more detailed as you look at them, let's consider Abstractions.

Although abstractions reduce viscosity (think style sheets in word processors) they require an abstraction manager, which will have its own set of CDs to think about (think of the Word style manager). Typically abstractions introduce their own type of premature commitment and viscosity.

With abstractions come the *Abstraction Barrier* and *Abstraction Hunger*. The Abstraction Barrier is the minimum number of abstractions that must be mastered before using the system; e.g. Java has a higher abstraction barrier than Logo, or a spreadsheet.

Abstraction Hunger on the other hand is a measure of the need or ability to define new abstractions. Abstraction-hungry systems can only be used by defining new abstractions (e.g. Java); abstraction-tolerant systems support but do not require new abstractions (e.g. some flavors of Logo), and abstraction-hating systems do not permit new abstractions, but often come with a few built in (e.g. most spreadsheets and graphics applications and diaries).

This analysis can be extended by the likes of the Attention Investment model of abstraction use or considering misfits between the system and user abstractions, but we'll stop the detailed analysis here, just noting as a passing comment that some of the most interesting developments are to do with abstractions in popular web sites and even in commodities - tagging systems, meta-data, personalised devices and even personalised cars.

Relations between activities and dimensions

Here is a brief and frankly unsupported description of the cognitive relevance of some dimensions to some activities:

	<i>transcription</i>	<i>incrementation</i>	<i>modification</i>	<i>exploration</i>
viscosity	acceptable	acceptable	harmful	harmful
hidden dependencies	acceptable	acceptable	harmful	acceptable for small tasks
premature commitment	harmful	harmful	harmful	harmful
abstraction barrier	harmful	harmful	harmful	harmful
abstraction hunger	useful	useful (?)	useful	harmful
secondary notation	useful (?)	-	v. useful	v. harmful
visibility / juxtaposability	not vital	not vital	important	important

Trade-offs

It's not easy to remove all possible user difficulties, but designers can exchange one kind of difficulty for another. The classic example is trading viscosity for abstraction hunger.

Resources

The main resource is the CDs Resource Page, maintained by Alan Blackwell:

<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/>

You will find a tutorial, a questionnaire system for evaluating, references to how CDs have been used in research and commercial development, etc.

The original paper: Green, T. R. G. (1989). Cognitive dimensions of notations. In *People and Computers V*, A Sutcliffe and L Macaulay (Ed.) Cambridge University Press: Cambridge., pp. 443-460

Attention investment: Blackwell, A.F. & Green, T.R.G. (1999). Investment of attention as an analytic approach to Cognitive Dimensions. In T. Green, R. Abdullah & P. Brna (Eds.) *Collected Papers of the 11th Annual Workshop of the Psychology of Programming Interest Group (PPIG-11)*, pp. 24-35.

A recent review: Blackwell, A.F. and Green, T.R.G. (2003). Notational systems - the Cognitive Dimensions of Notations framework. In J.M. Carroll (Ed.) *HCI Models, Theories and Frameworks: Toward a multidisciplinary science*. San Francisco: Morgan Kaufmann, 103-134.

Commercial usage: Clarke, S. & C. Becker (2003). Using the cognitive dimensions framework to measure the usability of a class library. In *Proceedings of the First Joint Conference of EASE & PPIG (PPIG 15)*.