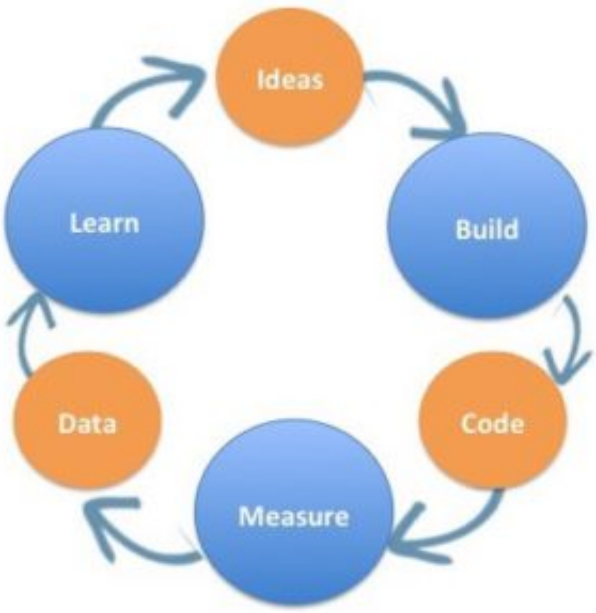
Why should technologists care about belief?

[cross-posted from Medium]

This is part of a talk on hacking belief systems using old AI techniques. The first part covered human beliefs, machine beliefs, and deep learning and the internet as belief-based systems.

Lean and Agile are based on belief



The build-measure-learn cycle. Image from https://steveblank.com/2015/05/06/build-measure-learn-throw-things-against-the-wall-and-see-if-they-work/

Most system development I see now uses agile principles: short, iterative cycles of development, and an emphasis on learning, preferably alongside the system's users, and adapting their systems to increase some measure of customer or user value. Lean is a similar idea at the company level, based around learning quickly about what works through prototyping and measurement (aka the "build-measure-learn" cycle). Lean Startup is the small-scale version of this; Lean Enterprise is the large-organisation one.

I'm going to focus on Lean Enterprise because it's more complex. In an established organization, Lean's usually a combination of learning and adapting existing business, and exploring new

business ideas (with innovations people bridging across these). Core ideas in this include purpose, customer value and incremental learning.

- Purpose is the ‘why’ of an organization, and acts as the common vision that everything in the organisation works towards (Daniel Pink’s videodescribes this well).
- Customer value is what the customer needs instead of the things we think they’d like to have. Concentrating on customer value reduces waste in the organisation.
- Iterative learning is about learning in small increments so we can adapt quickly when markets or needs change, and don’t bet too many resources on an idea that might not work out.

The build-measure learn cycle is at the heart of this. We start with ideas, build the smallest thing that we can learn something useful about them from (this might be code, but it might also be wireframes, screenshots, a pitch, a talk), test it, measure the results of our tests, learn from them, adapt our idea and iterate again.

And at the heart of this cycle are value hypotheses: testable guesses about the problems that we think we’re trying to solve (“our mailing lists are too general”), and the solutions that we think could solve them (“more people will vote for us if we target our advertisements to them”). We make each guess, often framed as a “null hypothesis” (a statement that what we’re seeing in our results is probably due to random chance rather than the effect that we were hoping to see), then try to prove or disprove it.

There’s a lot of design thinking behind hypothesis generation, and there’s a whole different post to write on how hypotheses breed smaller hypotheses and form hierarchies and structure, but the point here is that a hypothesis is an expression of a belief, and that belief and its management is at the core of Lean.

Lean value trees link beliefs together

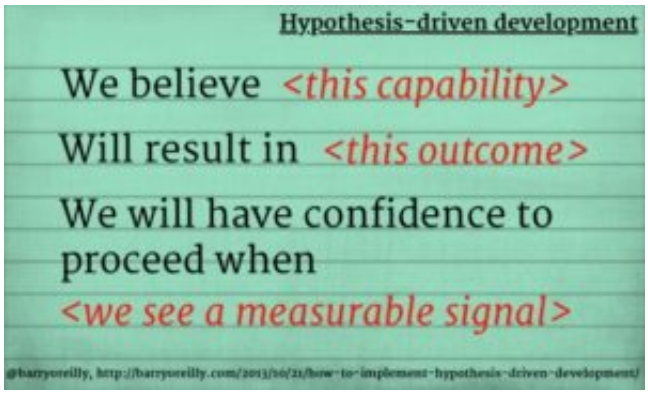
Mock-up Lean Value Tree for Bannon’s brain

A lean value tree (LVT) links an organisation’s purpose to its beliefs, metrics (the ways it

measures itself), hypotheses and experiments. The layers in the tree build on each other, and are:

- The organisation's purpose (aka mission): who is it, and what its vision is for itself. This is a stable belief, owned by the organisation's executive.
- Goals that support the purpose, framed as desired, stable and measurable outcomes.
- Strategic bets on those goals: different ways that we could satisfy a goal, with metrics associated with them. Stable in the medium term, owned below executive level.
- Promises of value: value-driven initiatives against those bets. Stable in the medium-term, owned by the product team.
- Hypotheses: measurable, testable guesses that support or refute the promises of value. Experiments on these are very short term (often quick and dirty) ways to learn about the problem space and potential solutions, and are owned by the development team.

Beliefs (hypotheses) are built into the tree, but belief is also built into how we manage the tree. One layer is called "strategic bets" because organisations rarely have the resources (teams, time, money) to do everything on their LVTs, and need to prioritise where they put those resources, by allocating them to the more 'important' things on the tree, where "important" is belief-based: a combination of beliefs in how big the market connected to a bet is, and how much value (financial or otherwise) is in that market.



HDD card (from Barry O'Reilly post on HDD)

Hypothesis driven development is an extension of the arc from test driven development to behavior driven development: its purpose is to learn, and to experiment quickly with ideas. Before a team tests a HDD hypothesis, it fills out a card (like the one above) describing the work done, the changes that we expect from that work, and a success condition (a "signal") that the team believes will happen if the experiment succeeds. Writing the success condition (e.g. "1% increase in donations") before testing is important because it a) forces the team to concentrate on what they

would accept as an outcome, and b) stops them fitting the test to the results they obtained (which could perhaps be called “optimism driven development”).

LVTs are useful models for talking not only about what we want within our own organisations, but also for testing our assumptions and beliefs about the goals and actions of other organisations, including organisations competing with us in the same markets.

We’re really optimising on other peoples’ beliefs

Lean practitioners talk a lot about customer value, and about including users in its feedback loops (either directly, or by tracking the changes in user behavior after changing the system). This optimizes systems not on an organisation’s beliefs, but on its beliefs about other peoples’ beliefs, filtered through those other peoples’ behavior. This is why things like good survey design and execution are difficult, with many types of observer effect (e.g. people doing and saying what they think you want to hear, responding differently to different ways of asking etc) that need to be allowed for.

Image: http://www.rosebt.com/blog/descriptive-diagnostic-predictive-prescriptive-analytics

But this isn’t just about understanding customer beliefs; it’s also about using that knowledge to predict and adjust customer behavior. Data scientists have a diagram for that, and talk about their work as a continuum from traditional statistics summarising data about events that have already happened (descriptive and diagnostic analytics), to predicting what might happen and how we could influence that (predictive and prescriptive analytics). But there is another box at the top of this descriptive-diagnostic-predictive-prescriptive diagram, describing a feedback loop where we learn and adapt our environment given that learning. This box is where the belief hackers, the roboticists and the lean enterprise people live.