I stumbled across TOGAF 8.x, wondered what it was because I couldn't figure it out from the website, and decided to pirate me a copy.  That's as far as I got.  
  
But I have an opinion anyways.  
  
TOGAF is a great way to make money.  That's why the standard from the "Open" group isn't free. It make money because you can read:

* http://en.wikipedia.org/wiki/TOGAF
* http://www.opengroup.org/togaf/ and it's friend http://www.opengroup.org/architecture/
* http://www.sparxsystems.com/ for certified software (I used Enterprise Architecture lite to produce some UML diagrams for an assignment once)
* http://www.builderau.com.au/strategy/architecture/soa/Is-Agile-development-secure-/0,339028264,339202460,00.htm (note URL has architecture & SOA in it)
* http://www.agilejournal.com/content/view/739/76/
* http://en.wikipedia.org/wiki/IEEE\_1471 (competing IEEE standard)
* and just about anything else on methodologies for tackling *huge* intertwined systems, with *huge* intertwined datasets

If at the end of reading all this, you feel like you're staring at an immense problem, but it seems that someone's thought this through in a clear, sure-fire, and concise way.  While everything you read seems really vague, you're left with a faint hope that there's magic inside that can save you from hard work and the whims of humanity.  
  
Sort of like the writings of L. Ron Hubbard.  
  
I haven't heard of a single large enterprise information system (CBA? Coles? Pacnet?) who's map of interconnects isn't a big bowl of spaghetti.  I'm becoming convinced that there's no silver bullet, that large == mess, but oddly enough a working one. No methodology in the world that's going to avoid spaghetti because one single person can't both understand the big picture, and master the still quite complex in their own right noodles that make it up, nevermind the sauce that interconnects it all.   
  
So what to do about these frameworks.

1. If you have zero morals, get certified & consult. Get paid for 3 months to write observe, document & suggest something, then move along. This pads your resume, ensuring a steady supply of work. Opens the door to being a CIO/CTO somewhere that's used to having things on fire everywhere.  (wtf didn't Majitek ask for checkpoints, hard deliverables?)
2. Gain a quick feeling for the standards that compete in the same space. i.e. TOGAF and the IEEE one (surely google+wikipedia will point out more).  This way in interviews, you can say that you've heard of the standard that employer X uses, and would approach the practitioners with an open mind.  Then change the subject & talk up what you've accomplished.

I recommend option 2.

---------------

After writing all that up, I read a wee bit more on the IEEE 1471 standard.  
  
Just compare & contrast the TOGAF page to the IEEE one.  
- togaf promises pixie dust / eternal life / etc  
- IEEE's seems to say: "you thought you were a software/systems guy in big complex environment, charged with making sense of it all & delivering functions {foo1...n}? nah, to succeed you'll need to know some psychology & sociology. The people that are paying you (aka stakeholders, many of them) all see what you're supposed to do, and what you're supposed to accomplish differently.  This should be apparent to you by now if you're seriously contemplating this role.  Anywayws, you have have to appease them without turning into a yes-man that bails on the project when lots of money is spent but nothing is done.  Here are some handy hints, and good luck, your odds of success are slim."  
  
ok, done now.