Is that Project Ready for You?

Open Source Maturity Modeling

Dave McAllister

Open Source? Why worry?

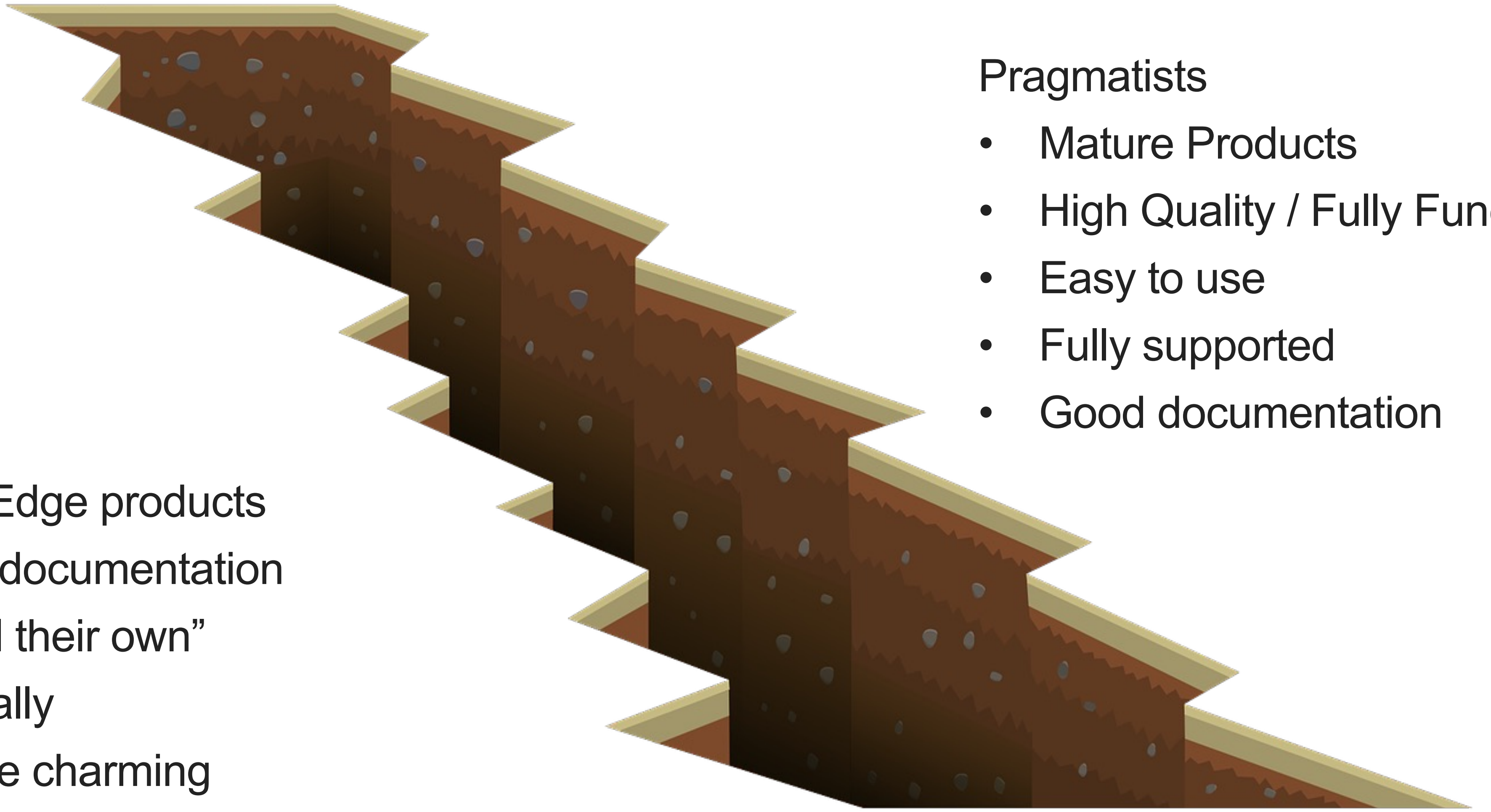
Open Source Software == Innovative, exciting, available

But

Is it ready for Prime Time?

- Is it stable?
- Is it supported?
- Is it advancing?

What’s Your Chasm?



Pragmatists

- Mature Products
- High Quality / Fully Functional
- Easy to use
- Fully supported
- Good documentation

Early Adopters

- Competitive Edge products
- Tolerate little documentation
- Willing to “roll their own”
- Support? Really
- Clunky can be charming

Maturity model to the rescue!

A brief history

1993 : Software Engineering Institute Capability Maturity Model

2003 : Open Source Maturity Model (OSMM) from Cap Gemini

2004 : Open Source Maturity Model (OSMM) from Navica

2004 : Methodology of Qualification and Selection of Open Source software

2005 : Open Business Readiness Rating (OpenBRR)

2007 : Open Business Quality Rating (Open BQR)

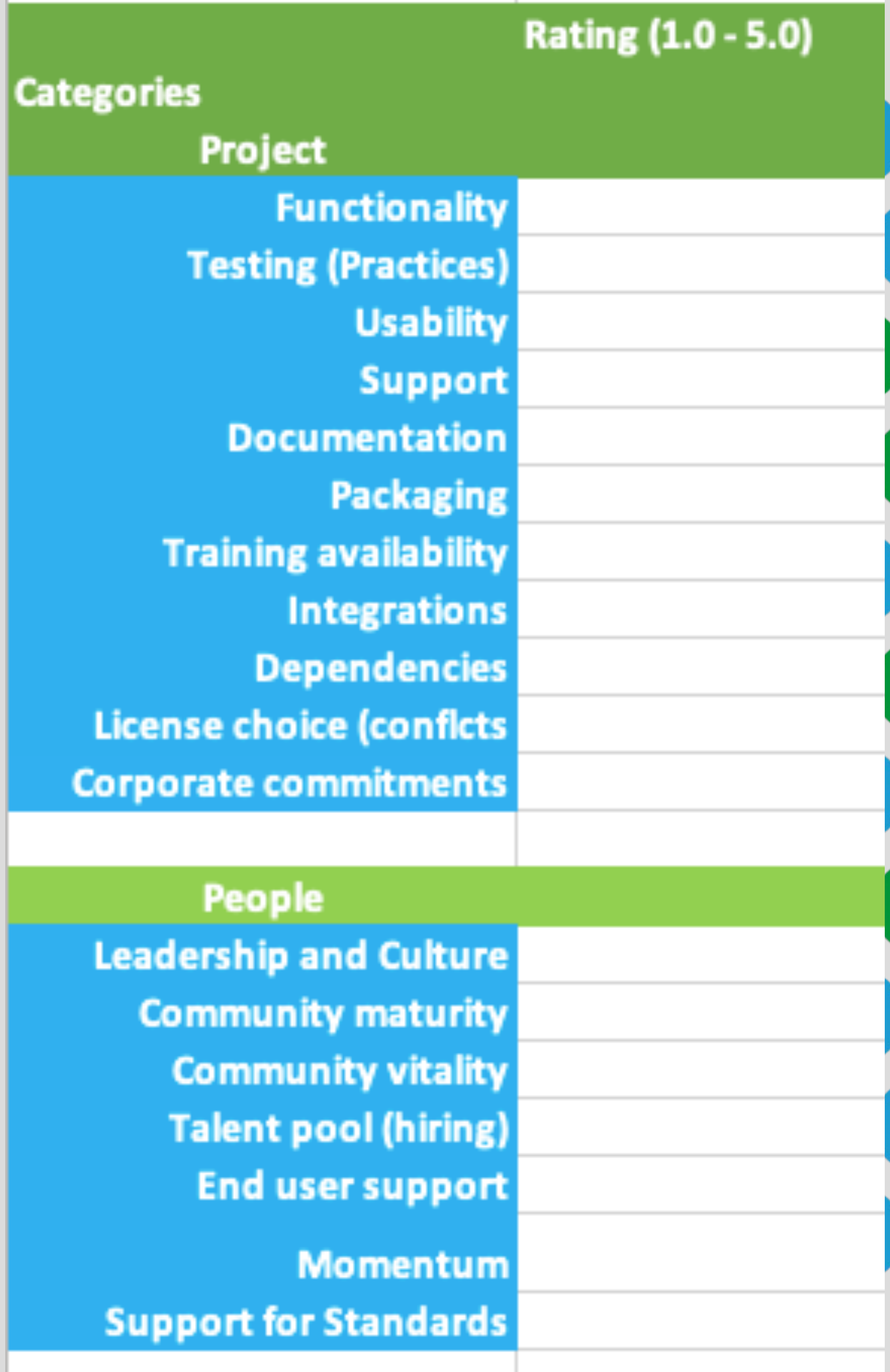
2008 : QualiPSo OpenSource Maturity Model (OMM)

The Common Elements

- Maturity models should describe
  - The Current State
  - The Desired State

Important Issues for consideration

- Functionality
- Quality Control and Assurance techniques
- Testing
- Risk Assessment
- Usability



Categories	Rating (1.0 - 5.0)
Project	
Functionality	
Testing (Practices)	
Usability	
Support	
Documentation	
Packaging	
Training availability	
Integrations	
Dependencies	
License choice (conflcts	
Corporate commitments	
People	
Leadership and Culture	
Community maturity	
Community vitality	
Talent pool (hiring)	
End user support	
Momentum	
Support for Standards	

Breaking it down

Three Assessments to make

- What are the product *elements*?
- What are your weighting factors?
- What is the overall score?

What matters to you

- Performance
- Scaling
- Usability
- Works correctly
- No license conflicts
- Easy to install
- Clear updates and roadmaps

Breaking it down

Three Assessments to make

- What are the product elements?
- What are *YOUR* weighting factors?
- What is the overall score?

How important is each piece?
How much do I need outside support/innovation?
Am I okay with just adoption?
Can I find people who already know this?
Will it be here next year?
Is it hot? (because it's useful)

Breaking it down

Three Assessments to make

- What are the product elements?
- What are your weighting factors?
- What is the overall _score_?

Item Score = Rating * Weight

Sectional Score = Sum (Item Scores) / Count (Item Scores)

Final Score = Sum(Sectional Scores)

Part 1: Product factors

- Packaging
- Training
- Integrations (to other products/projects)
- Dependencies on other products/projects
- Services

- Functionality (software)
  - Code
  - Design
  - Architecture
- Testing Practices
- Product/project support capability
- Docs
- Usability

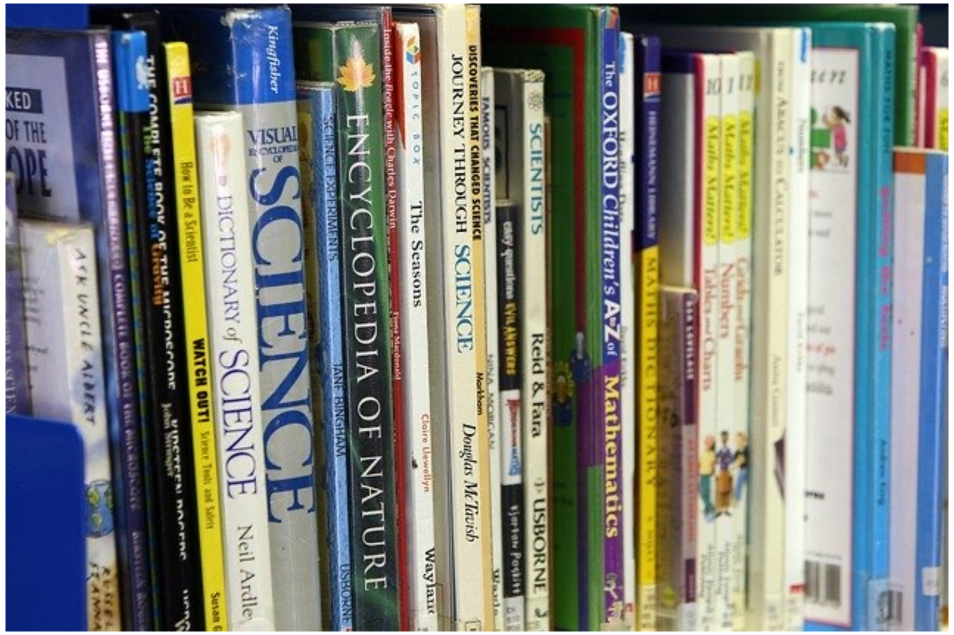
People (and other) Factors

- Momentum
- Support for standards
- License type and conflicts
- Corporate commitment (if applicable)

- Leadership and culture
- Community maturity
- Community vitality
- Talent pool
- End-user support

Assessment criteria

- Define your requirements
- Locate resources
- Decide availability
- Assign score



Your Weighting Factors

- What are your required factors
- How important are they?

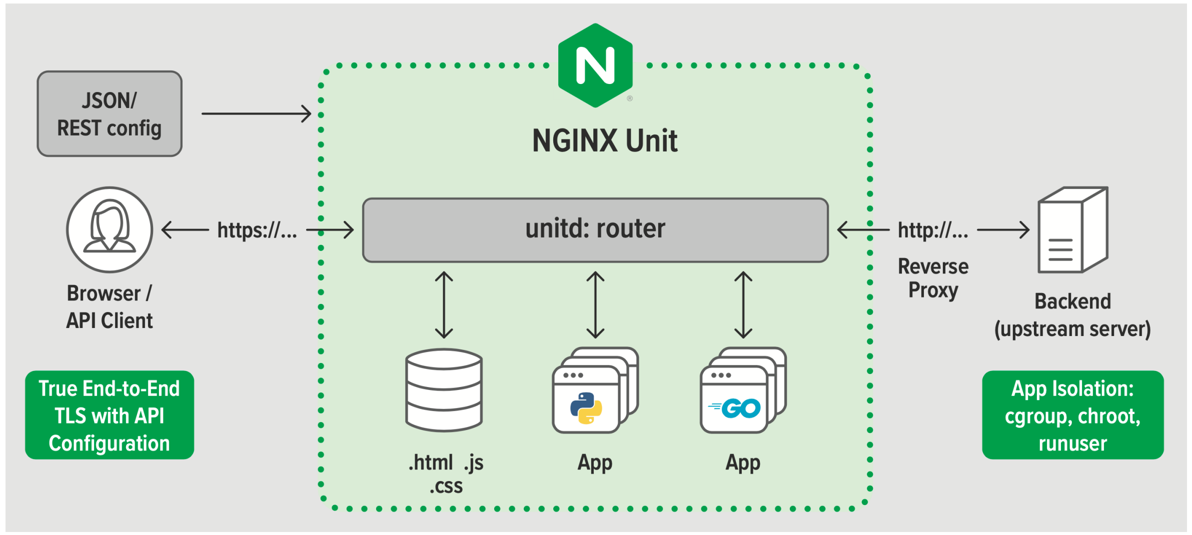
Example: NGINX Unit

simplifies the application stack

NGINX Unit

Open source Universal Web App Server

- Serves static assets
- Runs application code
- Proxies to backend



Sources

Github – https://github.com/nginx/unit

Mercurial

Website - https://nginx.org

SO and Reddit

Mailing list & archives

Changelogs

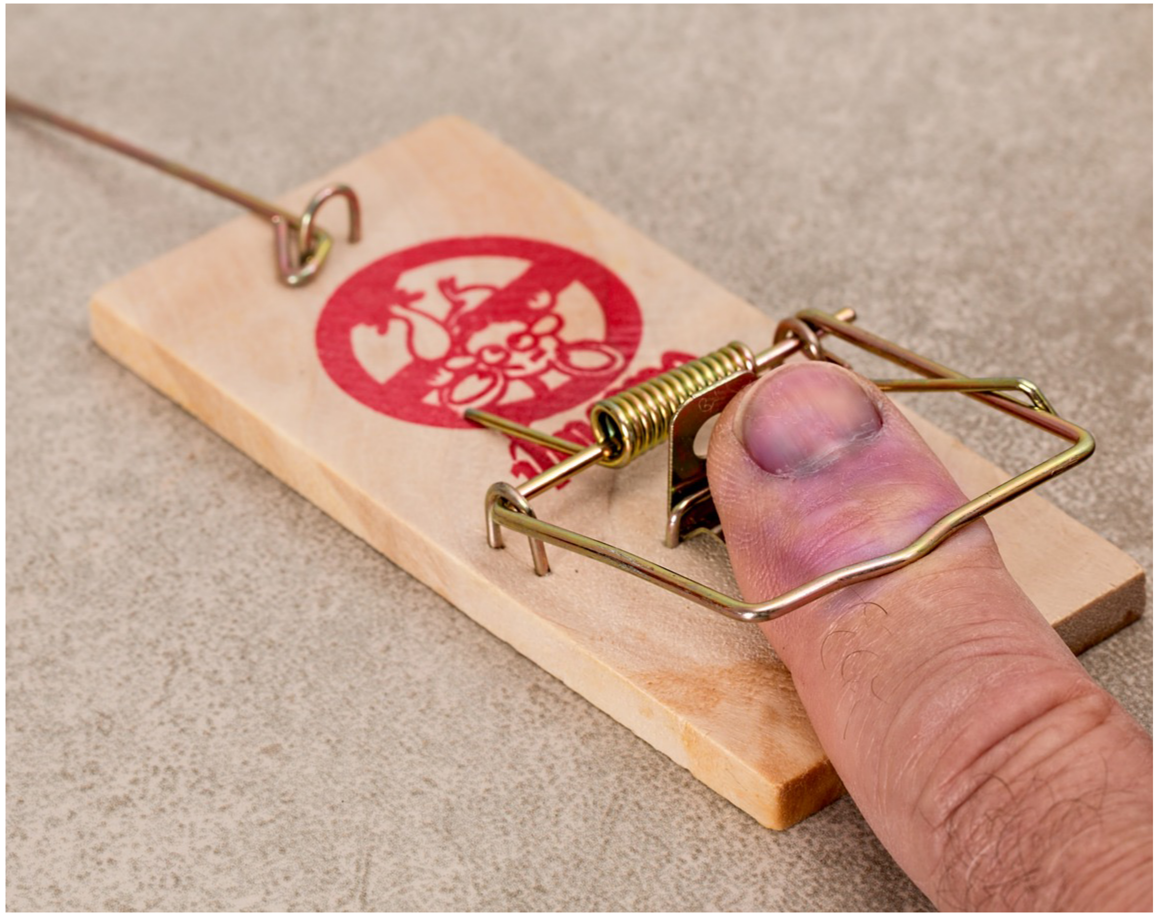
Other miscellaneous

NGINX UNIT OSMM

Categories	Rating (1.0 - 5.0)	Weight (1.0 - 5.0)	Score (1.0 - 5.0)	Power ranking	Notes
Project					
Functionality	4.1	4	16.4	82%	
Testing (Practices)	2.7	3	8.1	54%	no clearly defined testing processes
Usability	3.7	2.5	9.25	74%	
Support	2.5	3	7.5	50%	only commercial found
Documentation	3.6	3.5	12.6	72%	great docs, needs continual extension
Packaging	4	3	12	80%	
Training availability	2	2	4	40%	
Integrations	3.3	2	6.6	66%	
Dependencies	3	4	12	60%	
License choice (conflicts = lower rating)	4	4.5	18	80%	
Corporate commitments	4	3.5	14	80%	
		Sectional Score	**10.95**		
People					
Leadership and Culture	3	3	9	60%	NGINX leads. But no community
Community maturity	2.7	3.5	9.45	54%	E/U focus
Community vitality	2.8	3	8.4	56%	E/U focus
Talent pool (hiring)	1.7	2.5	4.25	34%	
End user support	2.5	1	2.5	50%	reddit, SO, other
Momentum	3.1	3	9.3	62%	
Support for Standards	3.7	4	14.8	74%	
		Sectional Score	**8.24**		
		Final Score	**19.19**		

So what's the catch?

- Maturity models oversimplify reality
  - But in overly complex cases this is useful
- Maturity models assume that there is only one true path
  - With open source though, that path may allow us to learn from those before us
- Maturity models presume there is an end state of nirvana
  - No nirvana, but are highly indicative of the current trends



Summing up

- OSMM can give you a framework for comparison
- OSMM can indicate the maturity of your own project
- There is no silver maturity bullet

And finally,

A tool is only useful when used appropriately. OSMM depends on you using it for comparison to other projects or to an acceptable risk point

Thanks for listening

https://wwww.linkedin.com/in/davemc