creating portable social networks with microformats

social networks

pownce.com
ma.gnolia.com
edenbee.com
upcoming.yahoo.com
last.fm
twitter.com
flickr.com

Social network fatigue?
More important than that: freedom of movement; reducing friction when moving from service to service.
This isn't about giving up one network in favour of another.
Interoperability is the goal.

username & password antipattern

Suck email addresses out of address books by requesting webmail username and password on a third-party site.
This teaches users how to be phished.
Insecure.

API+OAuth

google contacts
flickr

Safe and secure but complex to implement.
Does knowledge of someone’s email address really define a relationship anyway?

reuse
simple
existing

microformats

Principles of microformats:
Reuse existing standards (vcard, icalendar)
Simple by design. Don’t boil the ocean. Hitting 80% of use cases with 20% effort (Pareto principle).
Only encode what people are already publishing.
Don’t try to change people’s current behaviour (it’s really hard).
What format are most people already publishing in? HTML!

HTML

my details
my contacts
their details

Microformats are built into HTML rather than separate files
(out of sight is out of mind; invisible metadata rots).
HTML pages are RESTful. Can your website be your API?
Kind of... it's read only rather than read/write.
But look at the data that's already being published.

hCard

my details

Identifying people.

hCard

```
<span class=“vcard”>
<a class=“fn nickname url”
href=
“http://example.com/adactio”>
adactio
</a>
</span>
```

fn for formatted name, nickname for username, url for a web page that represents this person.
hCards do not need to be in-depth: there’s more to hCard than simply translating to vcard.
Don’t publish any more data than what you publish anyway.
There is semantic value in identifying a string of text as someone’s name.
There is no PERSON element in HTML. This is exactly what the CLASS element is for: adding your own semantics.

hCard

```
<h1 class=“vcard”>
<a class=“fn url” rel=“me”
href=
“http://adactio.com/”>
Jeremy Keith
</a>
</h1>
```

Profile page. This URL represents a person.

XFN

```
<h1 class=“vcard”>
<a class=“fn url” rel=“me”
href=
“http://adactio.com/”>
Jeremy Keith
</a>
</h1>
```

When linking to another URL representing the same person, use rel=”me”.
XFN is a proto-microformat (it didn’t follow The Process).

XFN

my contacts

XFN

```
<li class=“vcard”>
<a rel=“contact”
class=“fn nickname url”
href=
“http://example.com/username”>
username
</a>
</li>
```

XFN provides a number of values, but all you really need is rel=”contact”.
For a dating site like I’m In Like With You, you could maybe use rel=”crush” or rel=”sweetheart”.
For a professional site like Linked In, you could maybe use rel=”acquaintance” or rel=”colleague”.
But for most purposes, rel=”contact” is enough.
Beware of asserting rel=”friend” programmatically.

XFN

```
<ul>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
</ul>
<a rel=“me next”
href=“/contacts/2”>More...</a>
```

Pagination is a common pattern.
Use rel=”next” in combination with rel=”me”:
“This is another page representing this person.”

XFN

```
<ul>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
<li><a rel=“contact”... </li>
</ul>
<a rel=“me prev”
href=“/contacts”>...Back</a>
```

Likewise, use rel=”prev” in combination with rel=”me”.
REL values, like CLASS, can be space separated.

XFN

their details

XFN

```
<h1 class=“vcard”>
<a class=“fn url” rel=“me”
href=
“http://suda.co.uk/”>
Brian Suda
</a>
</h1>
```

Each one of your contacts has their own profile page, possibly linking off to another URL representing that person.

hCard & XFN

class & rel

That’s it! Sin é!
A few CLASS attributes and a few REL attributes and you’re good to go.

This isn’t a replacement for having a proper API: it’s a nice alternative for the simple stuff.
Allow access to data in as many formats as possible: XML, JSON, RSS, microformats.

parsing

code.google.com/apis/socialgraph

Parsing can be tricky. Spidering is even trickier.
Luckily Google provide the Social Graph API. Google does all the hard work.
Spiders XFN and FOAF relationships (FOAF relationships are translated to XFN).
Spiders rel=”prev” and rel=”next”, also spiders rel=”me”.

matching

fn

nickname

url

~~email~~

If you are have one social network and you allow people to import relationships from other social networks:
Try some fuzzy logic: a combination of nickname (username) and fn perhaps.
A matching url value is even better.
This won't get 100% accuracy but 80% accuracy with 20% effort is good.
You won't have access to email (often used a unique identifier). The other values should be enough.

what about...

trust?

walled gardens?

Trust: how can you believe these assertions? You can't. But that's true of everythin on the web.
It is not the job of formats to estabilish trust or authentication.
Protocol technologies like SSL and OpenID do that. Try mashing up OpenID with microformats.
Remember, microformats don't make any more assertions than what's already being published (they just add extra semantic clarity).

Walled gardens. Everything I've shown applies to sites that publish this information publicly.
You can still publish hCard and XFN on walled gardens but parsing will require some kind of authentication (maybe OpenID).

publishing

pownce.com
ma.gnolia.com
edenbee.com
upcoming.yahoo.com
last.fm
twitter.com
flickr.com

parsing

dopplr.com

getsatisfaction.com

the future...

one form field?

“Give me one URL that represents you.”
Let the system take care of the rest, discovering contact details and spidering relationships.

What you can do:
Add hCard classes and XFN rel values if you are already publishing a social network.
Use Google’s Social Graph API to allow users to import existing relationships.

design challenges

Make the sign-up process hassle-free.
Avoid jargon: don’t mention microformats or hCard or XFN.
Don’t import automatically. Allow users to accept or reject relationships.
Should you notify users when someone they know joins your social network?
Should you allow users to subscribe to, rather than import from, a different social network?

thank you

adactio.com

cc

microformats.org

go raibh maith agaibh