

The Case for a Location Metasystem

Nick Doty

npdoty@ischool.berkeley.edu

<http://npdoty.name>

Problem: ad hoc non-standards

- difficulty in developing location-based services (can't mix and match)
- compromising user privacy
- duplicating work in handling granularity

can compare this approach of addressing difficulty to that of “Hiding the Complexity of LBS”.

What is a metasytem?

- a *system of systems*
- *not* a single executable entity
- a set of protocols and principles

Microsoft's Identity Metasystem, which is meant to address the same problems of inconsistent (identity) systems, publishes a list of Identity Laws and a system of WS-* protocols.

Principles: Privacy

- consent
- minimal disclosure

Location systems must only reveal a user's location information with the user's consent.

iPhone dialog and Fire Eagle

Location systems should only reveal the location information necessary (for a particular application).

Fire Eagle good, iPhone bad

Principles: Privacy

- consent
- minimal disclosure
- retention
- re-transmission

Principles: Granularity

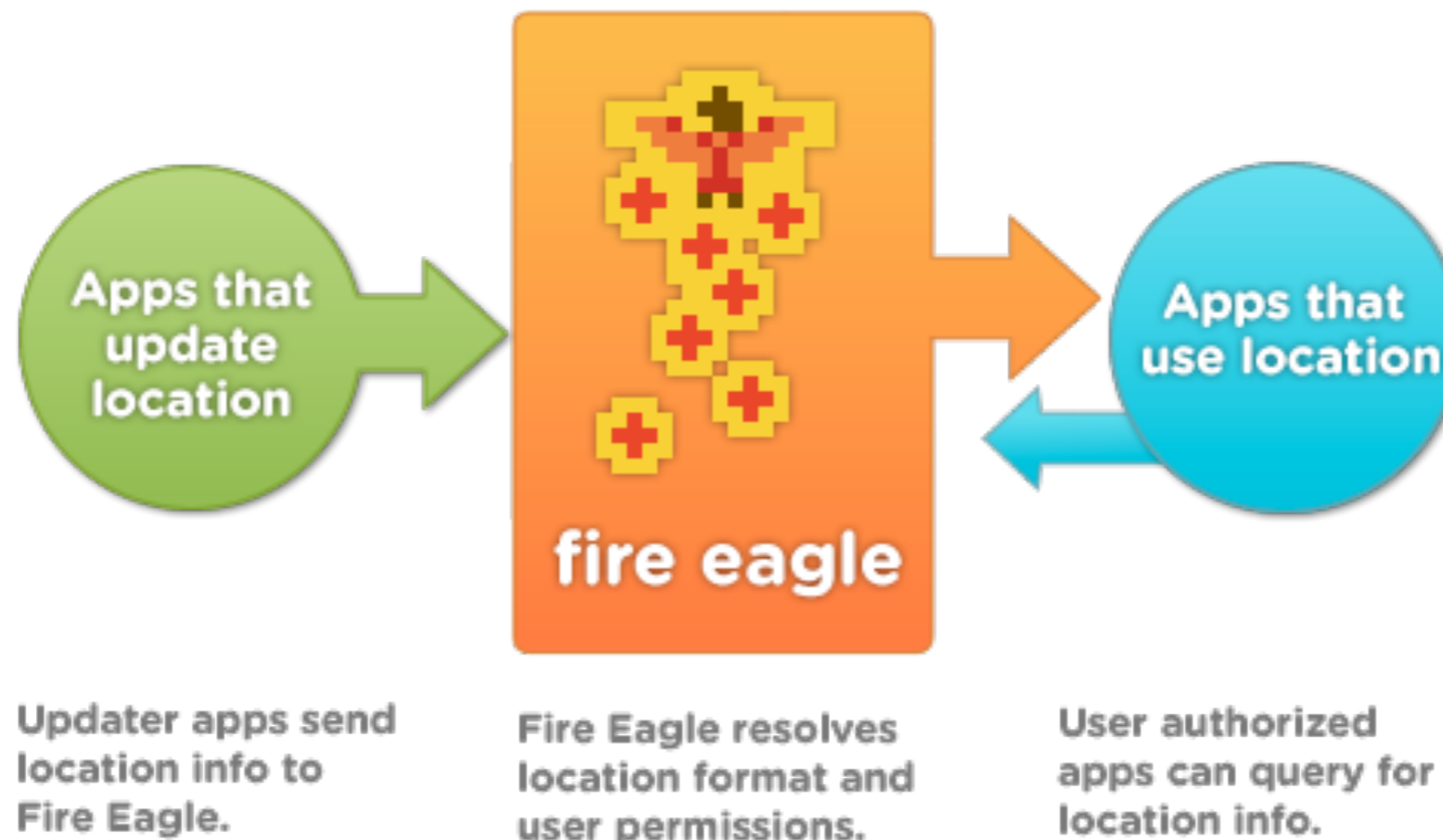
- both providers and consumers should specify the granularities they use
- necessary for minimal disclosure

Principles: Granularity

- both providers and consumers should specify the granularities they use
- necessary for minimal disclosure
- requires a standardized granularity hierarchy: *is that even possible?*

“semantic location” (not just user-created vocabularies, though that’s key too)
“Geographical Perspectives on Location”, Alistair Edwardes

Why a metasystem?



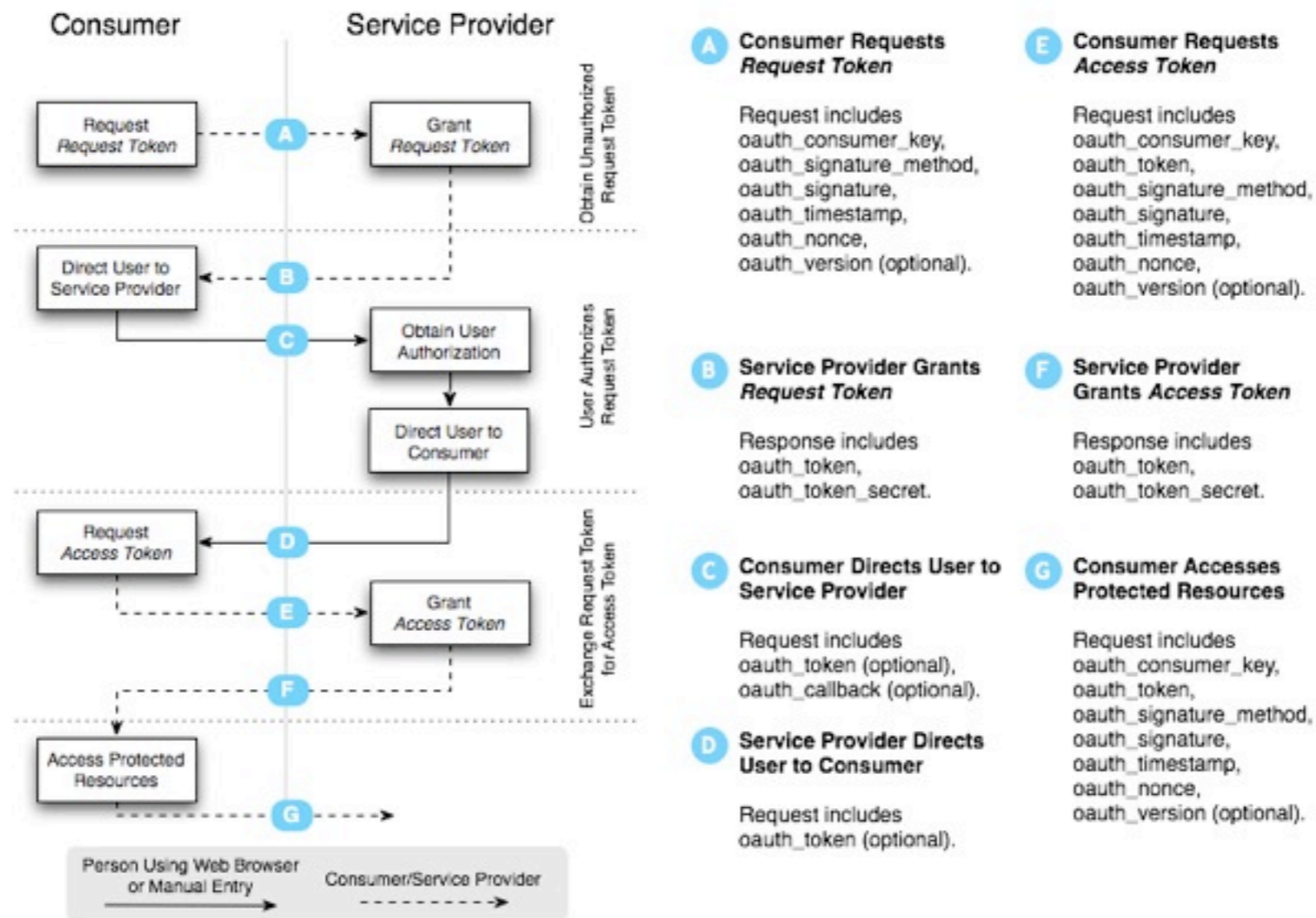
<http://fireeagle.yahoo.net/developer/documentation>

Why not just use Fire Eagle?

Microsoft's lessons from Passport.

Single point of failure; trusting a corporate entity; big target; no competition/differentiation

OAuth Authentication Flow



<http://oauth.net/core/1.0/>

OAuth as a model

Lightweight, and, in my personal opinion, fulfills some of the goals of the identity metasystem.

Oloc: an OAuth extension

```
requestLocationAuth<user, consumer, granularity>
```

```
requestGrantedCallback<user, granularity, token>
```

```
requestLocation<user, granularity[], token>
```

```
returnDataCallback<user, granularity, location>
```

Questions?