



# Building the Global Public Inclusive Infrastructure

**Colin Clark**

Co-chair, GPII Architecture Team

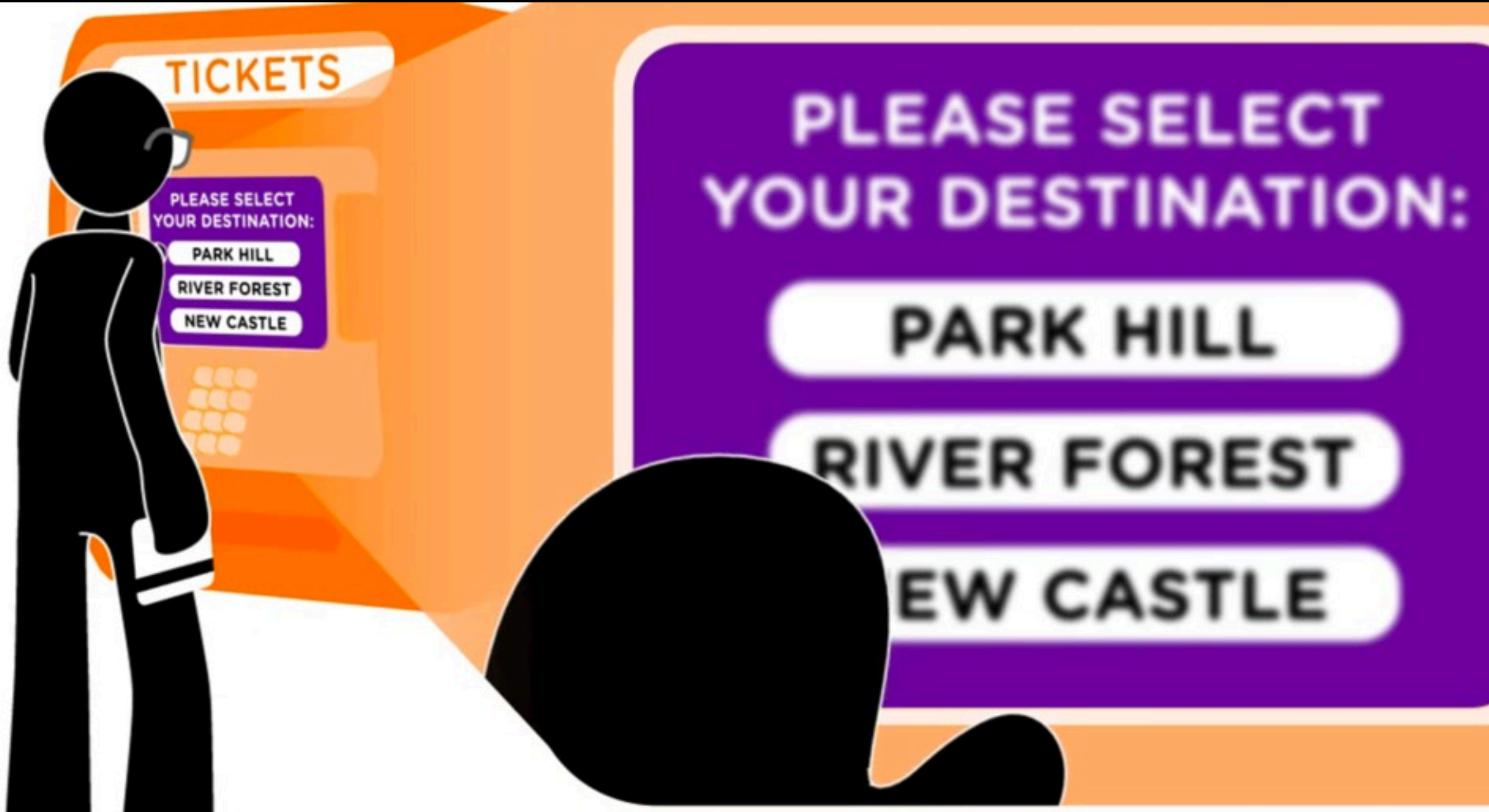
[fluidproject.org](http://fluidproject.org)



# Stuff we'll cover

- What's the GPII?
- How does it relate to our projects?
- The proposed architecture
- Roadmap and next steps

# What's the GPII?



# What's the GPII?

*GPII is a paradigm shift. The GPII will, for the first time, introduce automatic personalization of user interfaces and user context adaptation based on user preferences. Each information and communication technology (ICT) device will be able to instantly change to fit users as they encounter the device, rather than requiring users to figure out how to adapt, configure or install access features they need.*

Woah.

# What's the GPII?

- Automatic setup of assistive technology
- Adaptation of user interfaces
- Preferences stored *once* where users want them: out on the web or close to home

# The Technical Vision

- **Lower the cost of building accessibly:** Developers can draw from a diverse range of easy to find adaptive services
- **Foster innovation:** Novel assistive technologies delivered as modular services and components
- **Sustain** a flexible, personalized Web and beyond

- **GP11** is a big effort, an *outcome*
- **Raising the Floor** is the non-profit *organization* that coordinates the GP11
- **Cloud4all** is a European *project* to deliver parts of the GP11
- **Floe** is a *project* to deliver other aspects of the GP11 for open ed
- **Fluid** is a *community* that contributes to and sustains the GP11



# Desktop Use Case

- State their preferences **once**
- Go to a library and pick a computer
- Identify themselves *appropriately*
- Have the computer change according to their needs and preferences

**launch assistive technologies, setup built-in OS features, etc.**

# Web Use Case

ideas?

# GPII Architecture in a nutshell

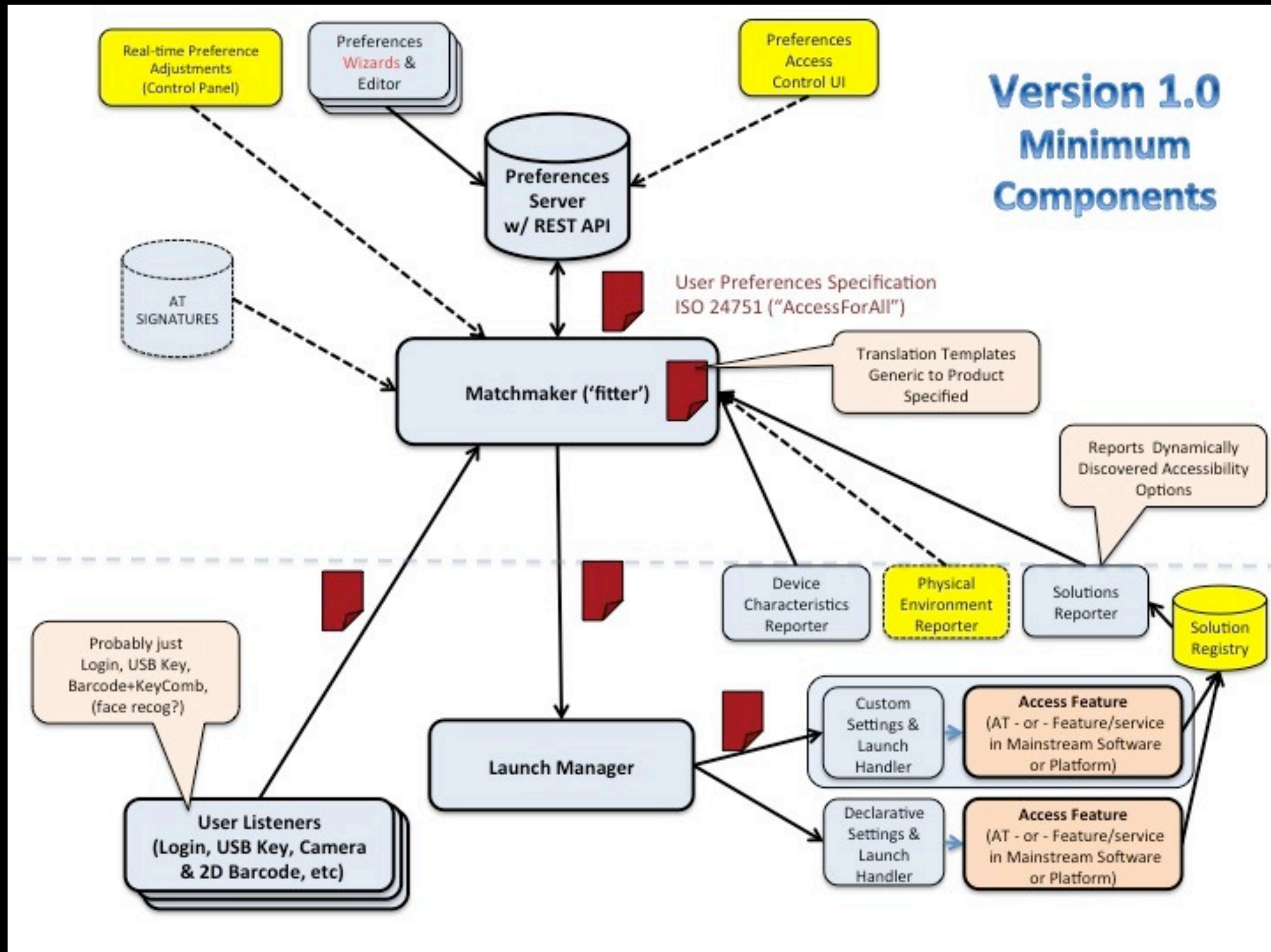
- Preferences (editors + server)
- Matching
- Standards & interoperability
- Development tools



# Technical Goals

- Language agnostic
- Relocatable
- Scalable
- Adoptable

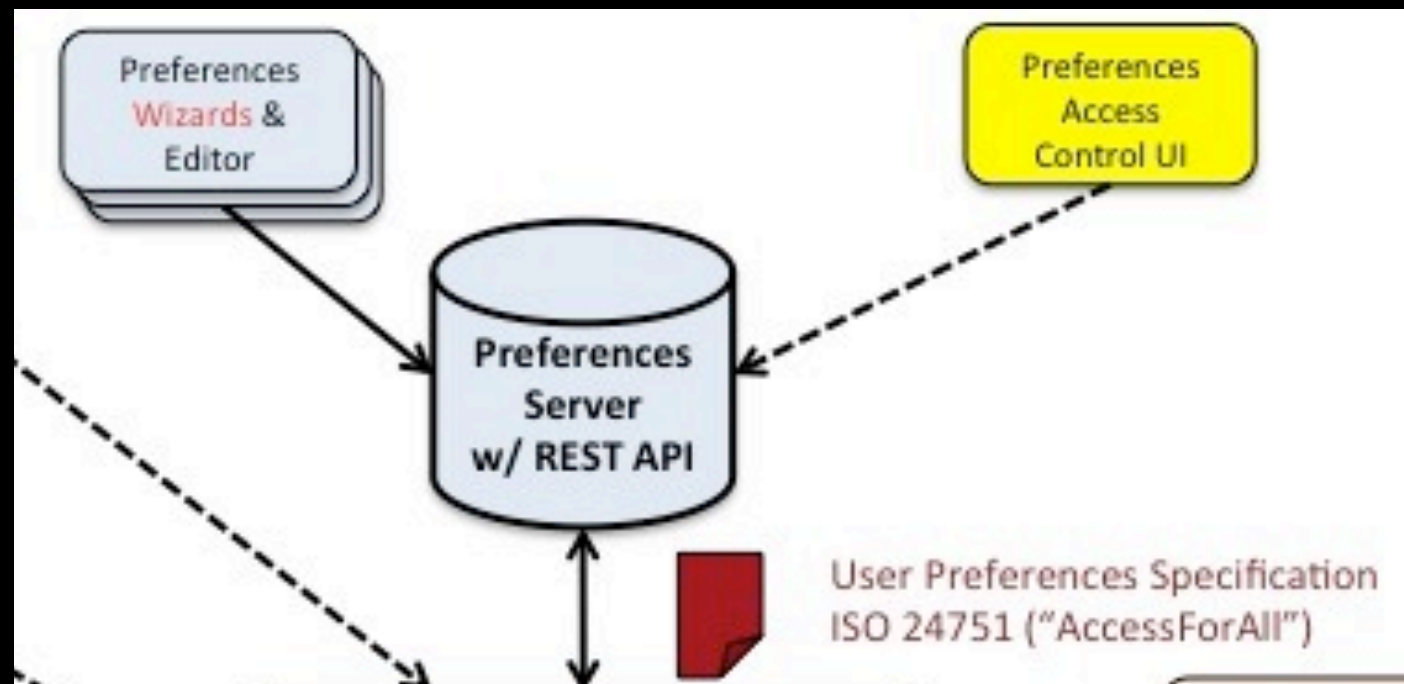
# Desktop Architecture



Woah.



# From the Top



- Preferences editor
- Web-based preferences server

# Preferences

## User Experience

- *Easy*: Speaks a language I understand
- *Ubiquitous*: I don't have to constantly reiterate or justify
- *Autonomous*: I can control who sees my preferences

# Preferences Editors

Support many preferences editors,  
designed for particular **audiences**,  
**contexts** and **needs**



# Preferences Editors

Support many preferences editors,  
designed for particular **audiences**,  
**contexts** and **needs**:

a **framework** for preferences editing

# Case Study: UI Options

A **TEXT AND DISPLAY** LAYOUT AND NAVIGATION LINKS AND BUTTONS RESET ALL

**TEXT SIZE** **A**  **A**  times

**TEXT STYLE**

**LINE SPACING**   times

**COLOUR & CONTRAST**

## Web Pages - HIDE


A web page or webpage is a *resource of information* that is suitable for the World Wide Web and can be accessed through a web browser. This information is usually in HTML or XHTML format, and may provide navigation to other web pages via hypertext links.

### Color, typography, illustration and interaction

[Web pages](#) usually include instructions as to the colors of text and backgrounds and very often also contain links to images and sometimes other media to be included in the final view.

### Elements of a webpage

1. Textual
  - [content with a textual representaion](#)
2. Non-textual
  - Static and Animated imagery
  - Audio
  - Video
3. Interactive



# Case Study: UI Options

TEXT AND DISPLAY LAYOUT AND NAVIGATION LINKS AND BUTTONS RESET ALL

TEXT SIZE: 1.4 times


TEXT STYLE: Verdana

LINE SPACING: 1.3 times

COLOUR & CONTRAST: DEFAULT

## Web Pages - HIDE

A **web page** or **webpage** is a *resource of information* that is suitable for the World Wide Web and can be accessed through a web browser. This information is usually in HTML or XHTML format, and may provide navigation to other web pages via hypertext links.



### Color, typography, illustration and interaction

[Web pages](#) usually include instructions as to the colors of text and backgrounds and very often also contain links to images and sometimes other media to be included in the final view.

### Elements of a webpage

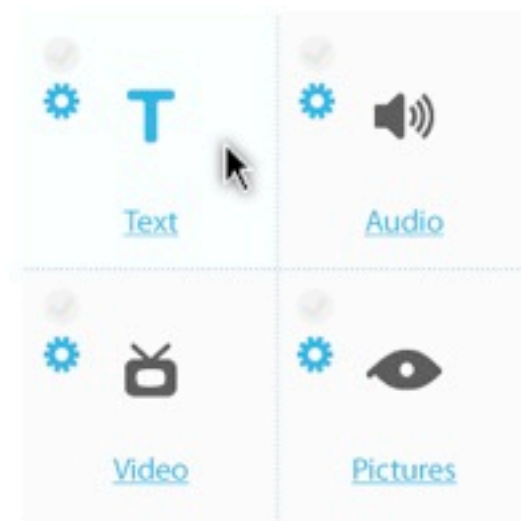
1. Textual



# ...Learner Options

Set content preferences for Modernist literature

I prefer **Modernist literature** with:



Summary of what's about to happen





We'll be presenting you **Modernist literature** in its original format.

NEXT

# ...Learner Options

Set content preferences for Modernist literature

I prefer **Modernist literature** with:

Additional text options	<input checked="" type="checkbox"/> <a href="#">Make it easier to read</a>	<input checked="" type="checkbox"/>  Text	<input type="checkbox"/>  Audio
		<input type="checkbox"/>  Video	<input type="checkbox"/>  Pictures

Summary of what's about to happen

We'll try to present **Modernist literature** to you such that it's largely made up of **text that's easier to read**.

NEXT

# ...Learner Options

Set content preferences for Modernist literature

I prefer **Modernist literature** with:

<input checked="" type="checkbox"/> ⚙️ <b>T</b> Text	<input type="checkbox"/> ⚙️ 🔊 Audio
<input type="checkbox"/> ⚙️ 📺 Video	<input checked="" type="checkbox"/> ⚙️ 👁️ Pictures

Additional picture options

- [Make it easier to see](#)
- [With detailed descriptions](#)


Summary of what's about to happen

We'll try to present **Modernist literature** to you such that it's largely made up of **text that's easier to read** and **pictures with detailed descriptions**.

NEXT

# ...desktop configuration

Control Preferences

 Web-4-All Preference Wizard Canada

**Control Preferences**

I would like an alternative to the standard keyboard

- Enhance the standard keyboard.
- Use an onscreen keyboard.
- Use an alternative keyboard.

I would like an alternative to the standard mouse

- Control the mouse pointer with keyboard.
- Use a mouse alternative.

I would like an alternative to the standard controls

- Use voice recognition.
- Use coded input (i.e. morse, chordic, quartering, or eight cell).
- Use word prediction.

# ...and a multitude of others

- Directed vs. free form
- Assisted vs. autonomous
- Making it more fun

Designed by skilled interaction designers, with users

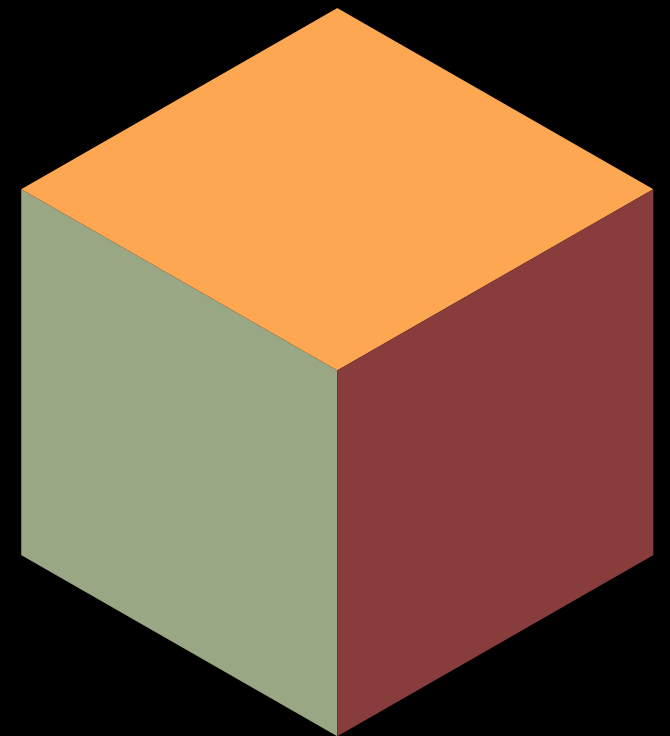


# Preferences Framework

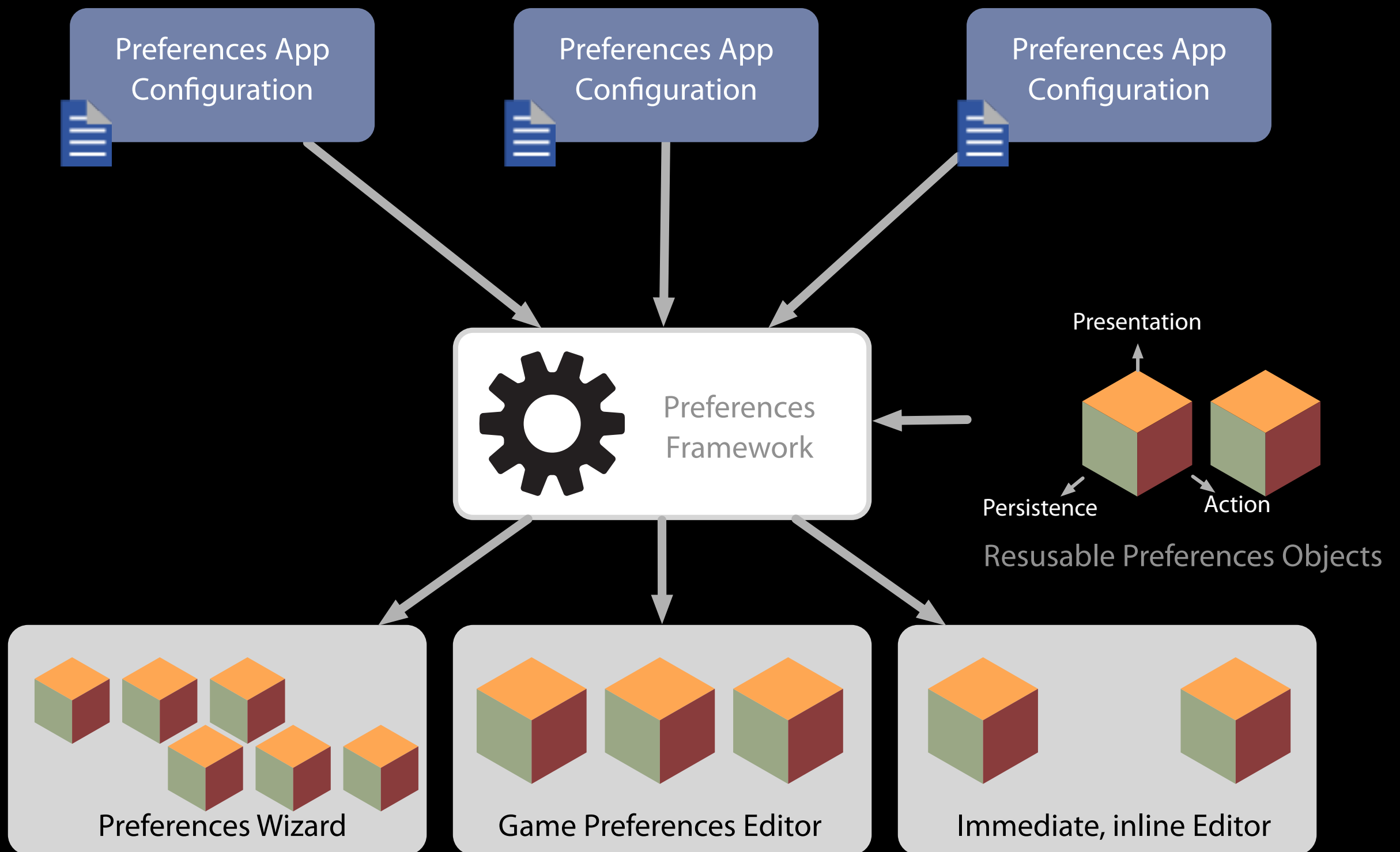
A three-dimensional model for preferences:

- presentation
- persistence
- activity

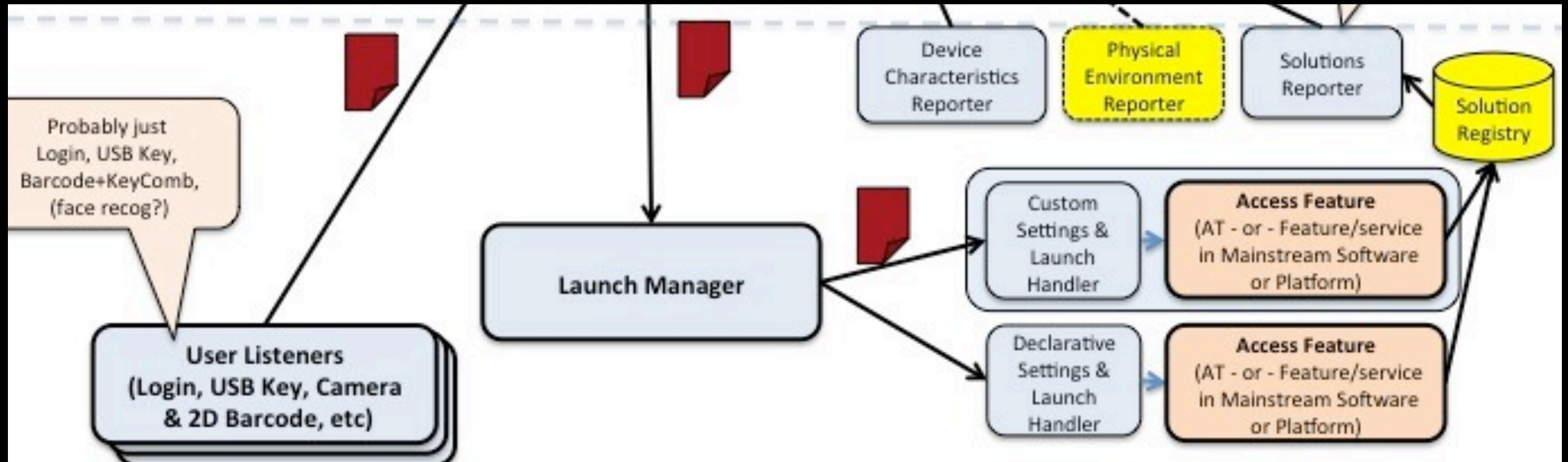
(how the preference is acted upon)



# Preferences Framework

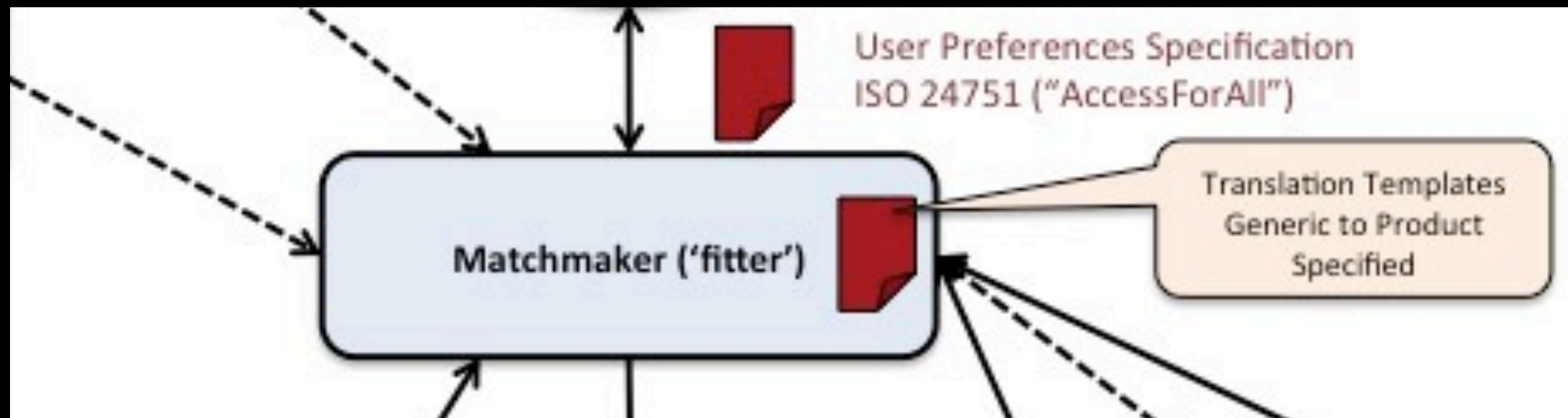


# And the Bottom



- A registry of all the “solutions” available on the computer
- A way to configure them
- Something that listens for the user

# In the Middle



- Something that can match the user's preferences with the available solutions
- Something to orchestrate it all

# Matching & Resolution

User Preferences

+

Device Capabilities  
& Software

=

Context

+

Environmental Data

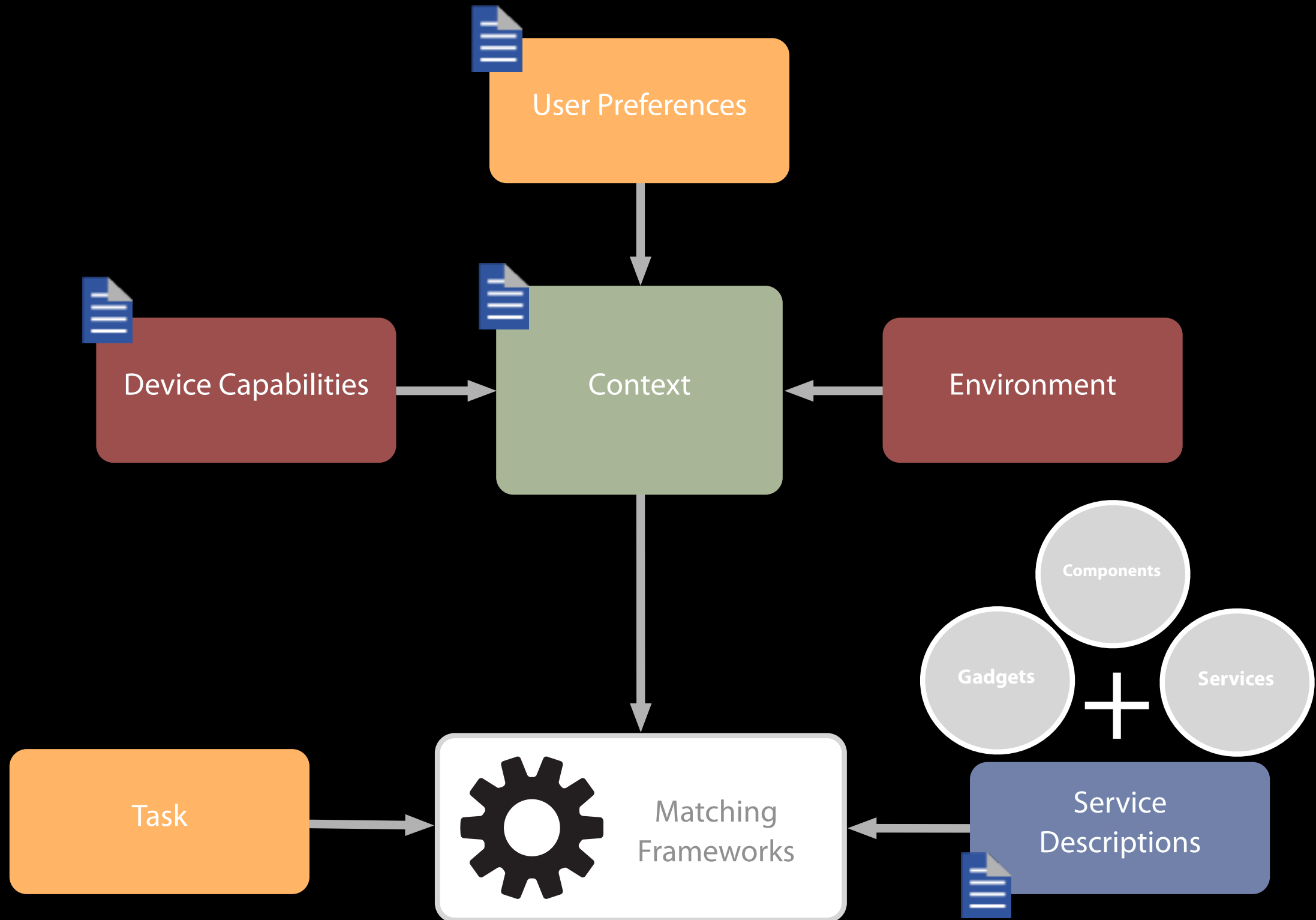
# Matching & Resolution

Context

*matched with* = Solution  
diverse user  
interfaces, assistive  
technologies, and OS  
features



# Matching & Resolution



# Matching

Declarative, interoperable formats for representing:

- Preferences
- Installed software and device capabilities
- Environment
- Context
- Service & UI capabilities

# Technical Challenges

- How do we make this work across platforms? (Linux, Windows, Mac)
- How do we make it scale to on-the-fly distribution?
- How about web-based ATs (e.g. WebAnywhere)?
- How does it work for web apps?

# Technical Solutions

- Use the web!
- HTML, CSS, JavaScript for user interfaces
- All modules and plugins built with JavaScript
- Don't make new custom APIs or use scary old ones like CORBA
- Use the web! REST and JSON payloads

# Technical Solutions

- Every module in the system is modelled as a URL
- Run an internal web server on the local desktop (no API-level difference between local and remote calls)
- Orchestrate the process using REST calls
- Every module consumes or produces standard JSON documents

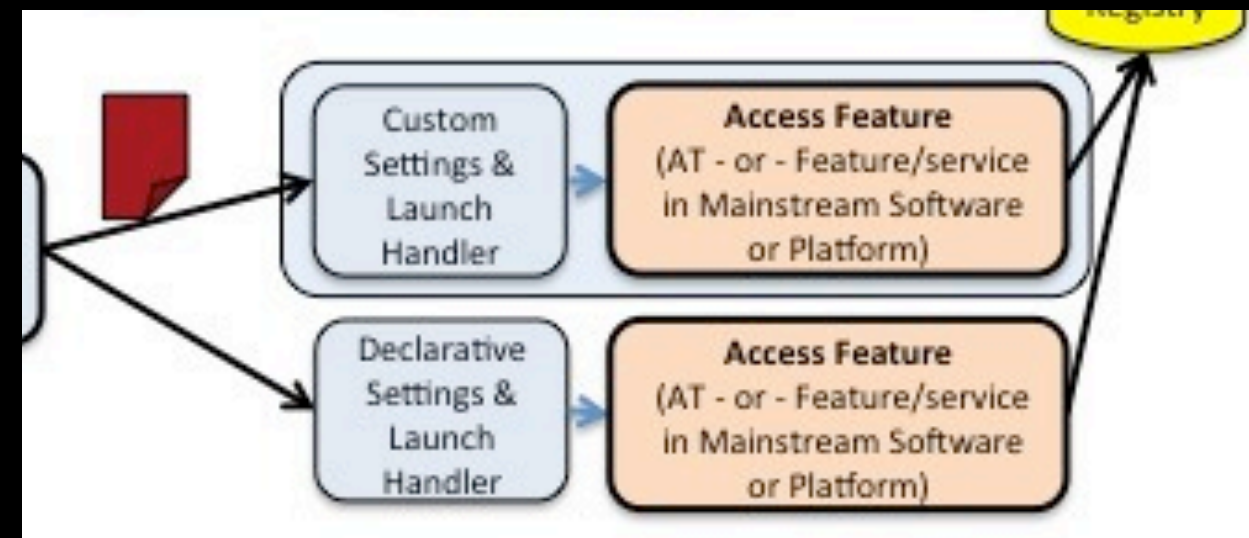
# Did you know Linux is awesome?

- A central store, designed to be accessed by third-parties, not just a single application
- Event-driven: setting changes, app knows about it immediately
- Oops, there are two of them: gconf and gsettings



# Configuring ATs

- Vendors need to plug their AT into the system
- AccessForAll deals with preferences in generic terms; apps need specifics
- Traditional model: write a plugin
- Risks:
  - Cost of code
  - Privacy leakage



# Configuring ATs

- We provide a set of typical settings handlers out of the box (e.g. for `gsettings` on Linux)
- Vendors provide a “settings map”
- The framework transforms from `AccessForAll` to specific `gsettings`
- Leverage what we’ve got: Infusion Model Transformation

# Roadmap

- Start with desktop
- Start with Linux and GNOME
- Build a sequence of small vertical slices
- Get something working end to end fast
- Do it with a small team

# Tools and Tech

- VM-heavy: automate using Vagrant & Puppet
- Node.js and JS for all cross-platform code
- Express, Infusion, and proto-Kettle
- gsettings and gconf: push other OS vendors to provide equivalent awesomeness







# Questions?

**Colin Clark**

e: [cclark@ocad.ca](mailto:cclark@ocad.ca)

t: @colinbdclark

[fluidproject.org](http://fluidproject.org)

