Reagent

Converting Ordinary Webpages into Interactive Software Agents

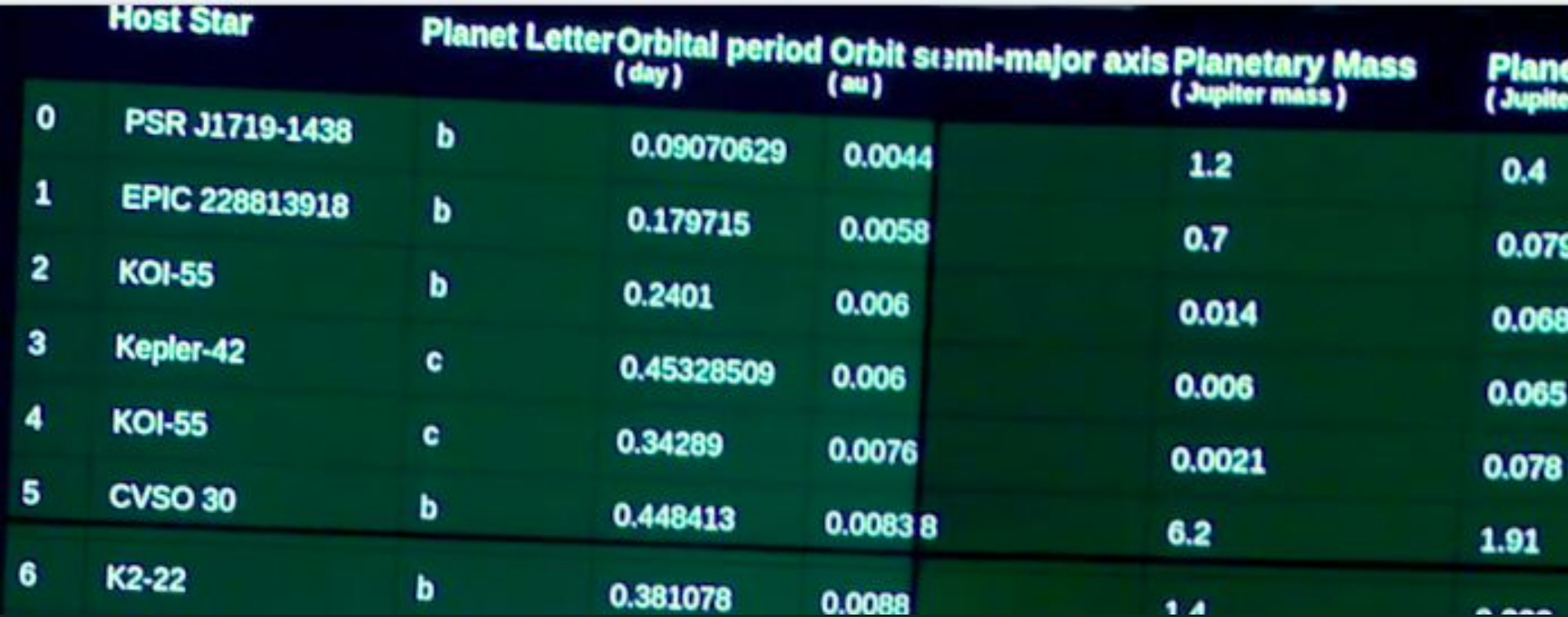
Matthew Peveler
Rensselaer Polytechnic Institute

Jeffery Kephart and Hui Su
IBM

IBM AI Systems Workshop
October 3, 2018

Motivation

- Voice activated assistants are growing in ubiquity
  - Examples: Alexa, Siri, Google Assistant
- Increasingly real world applications are being built around the idea of “smart rooms”
- Smart Rooms allow for interactions to consist of voice, pointing, and gestures
- User interactions are recorded and used for future interactions and disambiguation of intents
  - “What is that?”

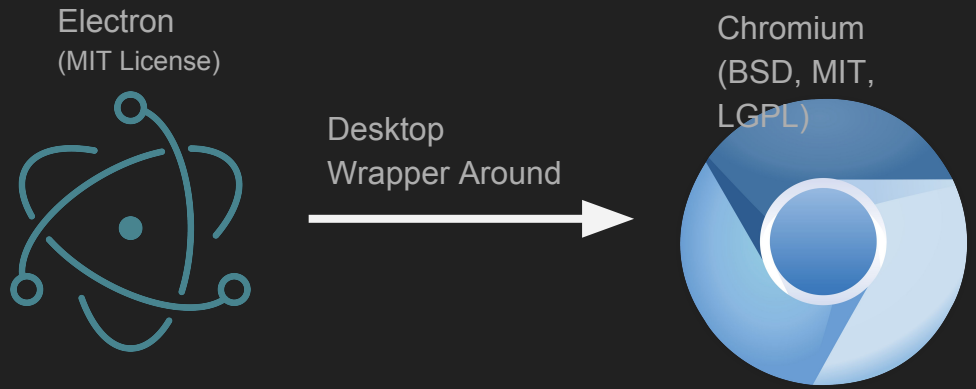


	Host Star	Planet Letter	Orbital period (day)	Orbit semi-major axis (au)	Planetary Mass (Jupiter mass)	Plane (Jupite
0	PSR J1719-1438	b	0.09070629	0.0044	1.2	0.4
1	EPIC 228813918	b	0.179715	0.0058	0.7	0.079
2	KOI-55	b	0.2401	0.006	0.014	0.068
3	Kepler-42	c	0.45328509	0.006	0.006	0.065
4	KOI-55	c	0.34289	0.0076	0.0021	0.078
5	CVSO 30	b	0.448413	0.0083 8	6.2	1.91
6	K2-22	b	0.381078	0.0088	1.4	

Kephart, J, Dibia, V, Ellis, J, Srivastava, B, Talamadupula, K, Dholakia, M (2018). A Cognitive Assistant for Visualizing and Analyzing Exoplanets. In Proceedings of Thirty-Second AAAI Conference. New Orleans, LA, USA.

Prior Work

- Work done by Kephart et al used Electron to display webviews (web pages) with content to users
- These webviews were built by his team where they could use JavaScript to capture various user interactions done via mouse
- These interactions were sent via websocket to central server to be usable within queries



The Problem

- Recording user interactions requires building out the webviews (as well as potentially necessary data sources) and special instrumentation in them
- Development of these webviews can be costly, time consuming, and a bottleneck
- Recording interactions with pre-existing web pages is problematic (the website owner probably won’t insert special code just for us)
- Web applications are increasingly popular and becoming the norm for usage over traditional desktop applications
- How to handle “single page applications” type web pages?
- How can we support multiple concurrent users?

Solution: Reagent

- Using Electron, we can pre-inject a JavaScript to run file on top of webviews
- This file waits until Electron fires a “dom-ready” event and then:
  1. Inject additional javascript files for initialization (i.e. socket.io) and then create socket between webview and Reagent server
  2. Scan and detect major HTML structures (e.g. tables) and uses specific injectors for those structures
  3. Reagent uses the injector to extract semantic information (e.g. column headers) and creates bindings onto these structures to capture user interaction
  4. Using a MutationObserver, Reagent watches for any changes to the page, firing off steps 2-4
- Can be used for any webview, whether internally or externally developed

Web Standards are actually Standard

- Many websites and governments are truly pushing for better accessibility:
  - W3C Accessibility Guidelines
  - Voluntary Product Accessibility Template (Section 508, US Govt)
- This in turn means sites are increasingly better about following “best practices” in web page structure
- Most browsers implement the same functionality/style leading to less “browser specific hacks” (like for IE)
- But the web still is messy in places, and Reagent can inject site specific injectors as necessary

Reagent is a transparent layer on top of pages

- JavaScript comes with a global object variable “window” that gets created on website load
- We can add a UUID to “window” to store Reagent functions and classes
  - window[reagent_id].function_name = function(...) { … }
  - window[reagent_id].class_name = class { … }
- We can add “data” attributes to analyzed structures to give them UUIDs
- We can transmit to the server anything that needs to be saved long-term (e.g. learned attributes)

Future Work

- Reagent gives us a bidirectional bridge between our system and webviews
  - Can get content at a given x, y pixel
  - Can trigger JS events on elements
- Can we utilize non-mouse interfaces to mimic mice, but allow multi-users?
- Build additional injectors (Chart.JS, D3, etc.)
- Integrate disambiguation with domains defined within Watson assistant

Thanks!

Matthew Peveler (pevelm@rpi.edu)

Demo video: https://bit.ly/2OGKvez