Robowaifu Design Document

>t. The /robowaifu/ Collective

April 14, 2020

Modularity

Things must be swappable and interface with each other

- Modularity will be really important.
- We need a library and protocol or at least a guideline for networking components together, both hardware and software.
- That way people can work on building different parts they find interesting and can afford to make, while others can quickly drop these components into their own robowaifu project.
- Good modularity will help ensure different components work well together.

Messaging I

Components need to communicate together properly

- Low latency, high throughput systems are needed.
- Will need to be able to pass large amounts of data around, such as AI tensor data and video data.
- Communications needs to be seamless across hardware and software boundaries.
- Similar to an IoT, but not on the Internet.
- Local data is stored within components, and shared with the rest of the internal robowaifu 'cloud' in a standardized way.
- Discoverable interfaces, so other components can easily find, query, subscribe to, and request what they need from each other.

Messaging II

Communications channels need to ensure integrity

- Where messages came from.
- Who modified messages.
- Where messages are going.
- Workaround communications bottlenecks, like the Internet does.
- Workaround rogue/misbehaving components that have been haxxored/damaged in some way.
- Workaround RF or other forms of interference.

Component libraries

Uniform software interfaces between components

- C extern ABIs for consistency and provision of bindings to other languages.
- This will enable loosely-coupled collaboration to proceed more smoothly.
- For example;
  1. One anon could develop a chatbot.
  2. Another could develop their own in another language.
  3. They could quickly interface the two programs to each other in a few lines of code and have them banter for fun.
  4. Another dev could focus on making a visual waifu program.
  5. Then another anon could combine everything together through the component libraries to create two visual waifus bantering with each other.
- All without the devs having to directly collaborate with each other on each other's project development.

Adaptability and Collaboration

The system must adapt to changing conditions and share workloads

- Components will need to be able to collaborate with other components.
- For example, a left-hand component will need a way to communicate with the right-hand component and coordinate their efforts.
- If a component becomes damaged or otherwise inhibited, the other components–and the system overall–should contain enough intelligence to adapt to this loss of component functionality.
- Even if all the pertinent data isn't immediately available, a component will still have to collaborate with other components. This means a component's current status must queryable by remote components.

No One-man Armies

Division of human labor will be one of the keys to success

- Anons have to find ways to work together with each other.
- Anons need to share the workload with each other.
- Having a reliable, modular system will allow each to contribute from their own interests.
- Let's have some fun with this!