

AI and the future of operating systems

Stéphane Bura

Chief Product Officer and Co-Founder, Weave.ai, 109 Cheapside, London, EC2V 6JS, UK

E-mail: stephane@weave.ai

Abstract. The current generation of Operating Systems (OSes) and apps are all about translating the interaction vocabulary to a smaller device and a touch screen. The next generation of OSes that will dominate the next twenty years of computing will be all about translating the apps and the interaction so that it leverages the AI capabilities of the device and that the AI leverages the data of the user. This paper will discuss the paradigm shift in operating systems.

Keywords: Operating system, OS, framework, user modeling, artificial intelligence, interaction

1. Why now?

Time spent working on desktop computers has peaked, and the upgrade cycle of software and hardware is slowing as it offers minimal marginal gains in terms of productivity.

Mobile devices are not about mobility, they are about convenience. They are usually the last computing device used at night and the first in the morning, often for performing work in bed. At the office they are used also when near the main desktop. Because of this, time spent on desktop and laptop devices has peaked and will never grow again, while time spent on mobile devices is ever-increasing. This is a major driver for the adoption of new technologies. There is a need for better productivity tools on mobile as today it's not possible to perform all work operations on mobile devices and as a result work is postponed until the workers are back at their desk.

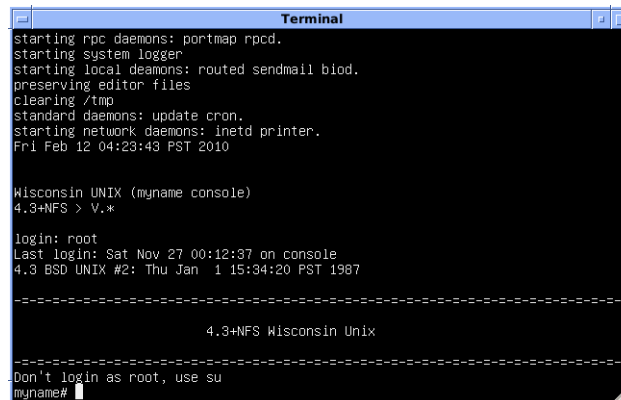
There is a correlation between screen size and personal productivity. People usually have multiple applications open at the same time so data from one app can be copied onto another while reading streams of communications data.

When transitioned to a mobile device the display surface a user is reduced from a >20" screen to <5" and the input mechanism goes from a full keyboard and a mouse/trackpad to a thumb. It is because of these form factor limitations that the leading productivity apps are single purpose. And the most successful ones are very specific at solving one task in the simplest way possible as opposed to having an app with an extensive set of features.

As a result, the average productivity worker has an extensive number of apps, each for a single purpose: email, algorithmic emails, calendars, organizing meetings, instant messaging, etc. Furthermore different apps are used in different contexts, so that a mobile device might have multiple instant messaging apps, one for family, one for friends, one for work, and so on.

This causes a multiplication of functions that could be provided by the OS (operating system) and a cluttering of the notification system that has no understanding of the data that it is processing.

At Weave.ai we believe there is a better way.



```

Terminal
starting rpc daemons: portmap rpcd.
starting system logger
starting local daemons: routed sendmail biod.
preserving editor files
clearing /tmp
standard daemons: update cron.
starting network daemons: inetd printer.
Fri Feb 12 04:23:43 PST 2010

Wisconsin UNIX (myname console)
4.3+NFS > V.*

login: root
Last login: Sat Nov 27 00:12:37 on console
4.3 BSD UNIX #2: Thu Jan 1 15:34:20 PST 1987

-----
4.3+NFS Wisconsin Unix
-----
Don't login as root, use su
myname#

```

Fig. 1. First it was the command line (1960s).



Fig. 2. Then it was replaced by Windowing systems (1980s).

2. The evolution of operating systems

Every twenty years there is a paradigm shift in operating systems where the previous computing layer is abstracted and a new form of computing is adopted that gives leverage to operations (see Figs 1–3).

3. Artificial Intelligence-first Operating System

The rapid evolution of artificial intelligence technologies over the past five years coupled with the push to increase productivity on mobile devices create the groundwork for transition into a new computing paradigm. Conceptually it's a *contextual card system*. Previous attempts like Windows Metro and Google Now are notable examples, and it is our belief that Windows Metro was closer to what could have been a true next-generation Operating System (see Fig. 4).

Metro was an empty User Interface where the cards were simple representation of existing folders and did not provide any advantage over a more traditional display, but it had the drawback of changing

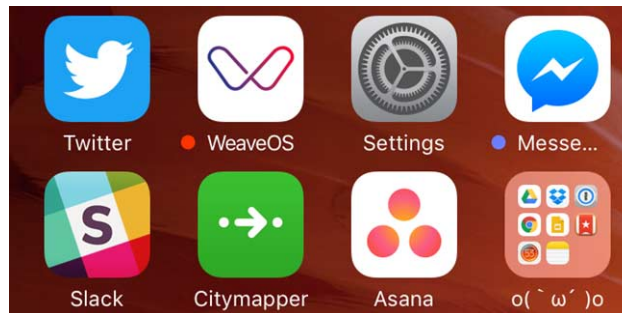


Fig. 3. Then it migrated to an App economy (2000s).



Fig. 4. The future looks a lot like Metro.

a user workflow. Additionally there were no predictive or contextual API's or developer tools that took advantage of the new design. *In short it was an interesting design concept, but there was not enough artificial intelligence to deliver a next-generation experience.*

4. Characteristics of an Artificial Intelligence-first Operating System (OS)

The next logical evolutionary step is an AI-first Operating System. It is a challenging goal that requires multiple steps, but the key advancements in technology that define a next-generation OS are:

1. Contextual cards and disappearing apps;
2. Extensible semantic framework;
3. User modeling;
4. Artificial Intelligence grammar.

4.1. Contextual cards and disappearing apps

The main interaction with a new OS would be a card system where each card can represent a micro-format with dynamic data, contextual intention and a visualization of said data. The cards would be 1) ranked by importance to the user and 2) by the intent based on historical data pattern of a specific context.

The shift here is going to be the disappearance of apps in a traditional sense so that a card can be “*your next meeting*” and it can include people, places, and documents related to it. Based on the intention of the

user, the card can then invoke a transit module (roughly equivalent to today's Citymapper app) to navigate to destination or a communications module to chat with the meeting participants and automatically alert them of a possible delay (again pulled automatically from the Citymapper API). These modules are a new kind of interface between activities, carrying context and intent across your device's features. The key difference is that the data of the card is held by the device and that a hypothetical transportation app is a User Interface visualization of that data with some specialized API processing.

This implies that apps as we know them are bound to disappear and the full product experience will happen inside a card or its notification (again, the boundary between the two becomes fuzzy as some products could be experienced as notifications, especially as the screen size is further reduced by Augmented Reality (AR) and smartwatches). *Key takeaway: the card becomes the destination, not the app, because the card is tied to an intention.*

4.2. Extensible semantic framework

Navigation apps know what a location is, email apps know what files or contacts are and so on. The knowledge representation of semantic data today is app-specific and only few dimensions are described in the OS (time and space usually). In an AI-first OS the model is turned on its head and the OS has an internal knowledge representation and the app derive their data structure from it.

This enables different apps to communicate natively (right now your app can call an Uber only because its developer manually added the Uber API or uses a 3rd party like Button) and also has the side effect of enabling an OS to invoke multiple apps to work on a problem. Taking this further, it enables the user to express desires "going out for dinner" and the OS is able to create a plan and have different apps to satisfy each step of the plan. Furthermore, since the required functionalities are defined by the user's intentions as expressed in the framework's language, the OS could automatically look online in a "feature store" for a suitable functionality if there were none on the device.

Google Now is not able to understand what an object is unless Google adds a card about it specifically in its system (like for cricket or basketball). To avoid pitfalls like this the system needs to have an extensible semantic framework in which the OS has a number of primitives that can be used by app developers to describe new objects and expand the interaction language. Semantically describing the world is a task bound to fail if it rests with the OS creator as the edge cases are almost infinite. *Key takeaway: the OS understands all the data that is manipulating and passes that knowledge to the apps.*

4.3. User modeling

The other big trend that is already happening is the move to an OS that is modeled with the user's data. To some extent there was always a level of customization in OSes, whether scripts or color preferences, but the next step is the system reading the user's data, understanding it, classifying it, correlating it, and then responding on the basis of it.

The user modeling is at the core of a set of predictive APIs that need to be available to developers, such as "what next" (which itself is a container for multiple possible options depending if the next action is travel, communications, reading etc.). The new Google Awareness API is an example of this. Furthermore, this generic API would allow user feedback across all activities, since the OS could explain why its making a recommendation and the user could highlight or reject some of the criteria. This would turn what it today a passive activity (receiving a recommendation) into interactive query building, bound to be much more relevant to the user's needs. *Key takeaway: the OS will be able to understand the user, its uniqueness, and be aware of the user's current activities.*

4.4. Artificial Intelligence grammar

All of the previous key technologies described here coalesce in what is going to be the litmus test of a next generation AI-first OS: an agnostic Artificial Intelligence grammar. By AI grammar we mean a set of operators like What, When, Where that can be applied to data. Some of this grammar is already part of our daily life: every time we search on Google we are performing “What” + our search query. With the introduction of mobile devices and calendars, “When” and “Where” can be answered. A lot of progress is being made in extracting information and making plans so that “How” can be answered.

But all of these operators are performed by a black box where the user has to accept the results of the AI. While we do not think in terms of “Why” when issuing commands to an AI it becomes apparent when the query fails and we are left unable to explain to the system “Why Not?” With such architecture the user can explain itself to the system and a failed query is turned into training data.

But it is the combination of all these technologies that will deliver a truly transformative next-gen experience and give users a device that can understand their desires and help fulfill them.

This grammar doesn’t need to be verbal. It can be mapped onto an NLP system as well baked into the contextual card system. To some extent it will be secondary how the information is surfaced to the user once the system can handle a grammar for communications. *Key takeaway: the OS needs to have a symbolic AI grammar and “Why” is the operator that will unlock AI communications.*

5. What is the role of Weave.ai?

At Weave.ai we are developing WeaveOS, a suite of artificial intelligence technologies that can power a next generation operating system. We have years of expertise in research, AI architecture and user-centered design. If you are interested we would love to discuss this further. Contact us at: stephane@weave.ai/rodolfo@weave.ai.

About the author

Stéphane Bura has been working in Artificial Intelligence and interaction design for more than twenty-five years. He has been the creative director of a video game studio and consulted on game design for companies all over the world. He co-founded two AI start-ups, both with the goal of transforming the relationship users have with their devices: Storybricks, that built tools for game developers to generate emotional connections in players, and Weave.ai, that proposes a new model for connecting data, contexts and interactions.