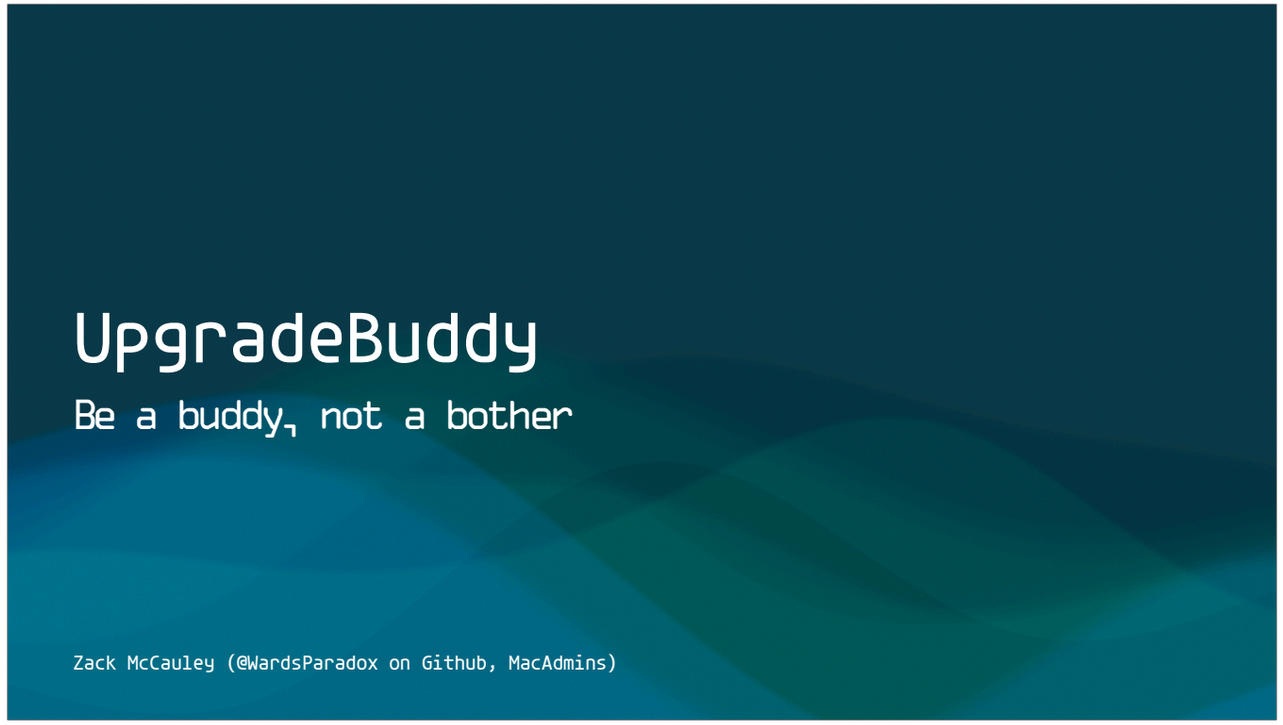


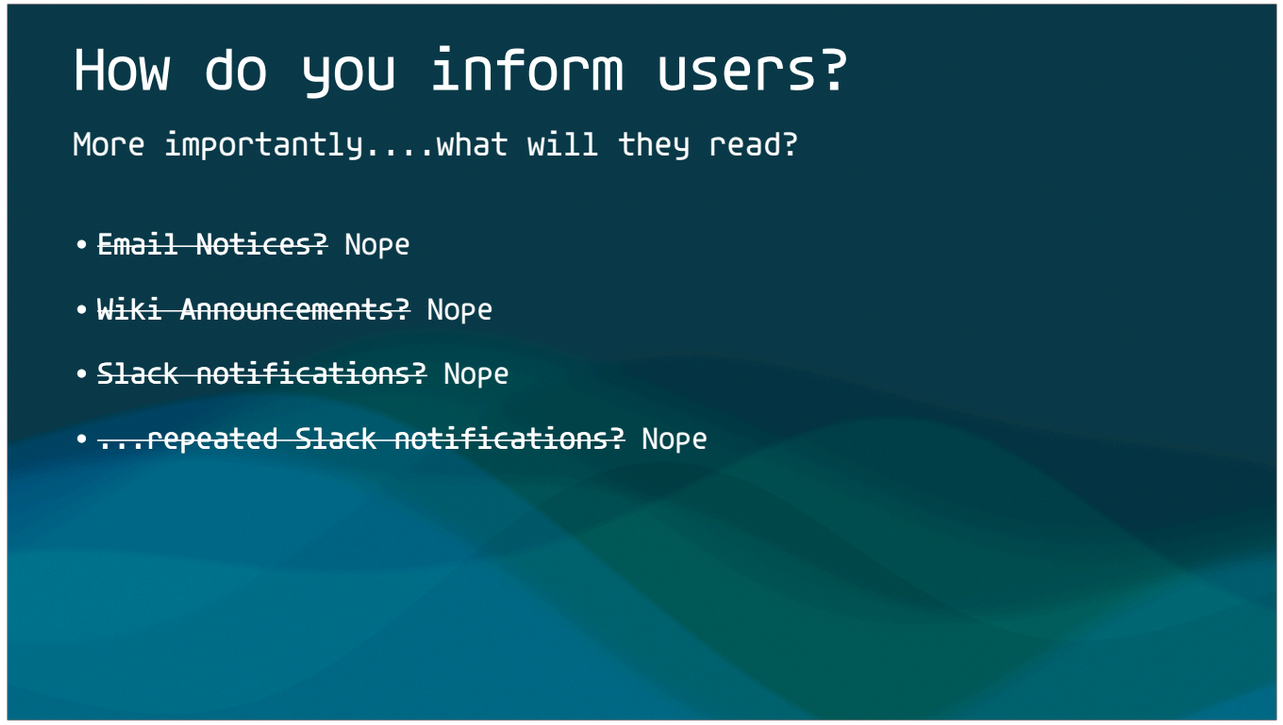
Good Morning, soon to be afternoon! My name is Zack and I'm a Systems Engineer at Amazon where we manage many many machines.

I'm sure most of you have had some sort of OS change that required some manual intervention by your users. A great example of this was with macOS Ventura. Apple upgraded to openssh 9.0 which also included a change where SHA1 keys no longer worked by default, instead replaced by a newer modern ECDSA key.

So given this sort of change...

How do you contact your users about a change that may impact them at upgrade to a new OS?

Depending on your Org size... an option may be:

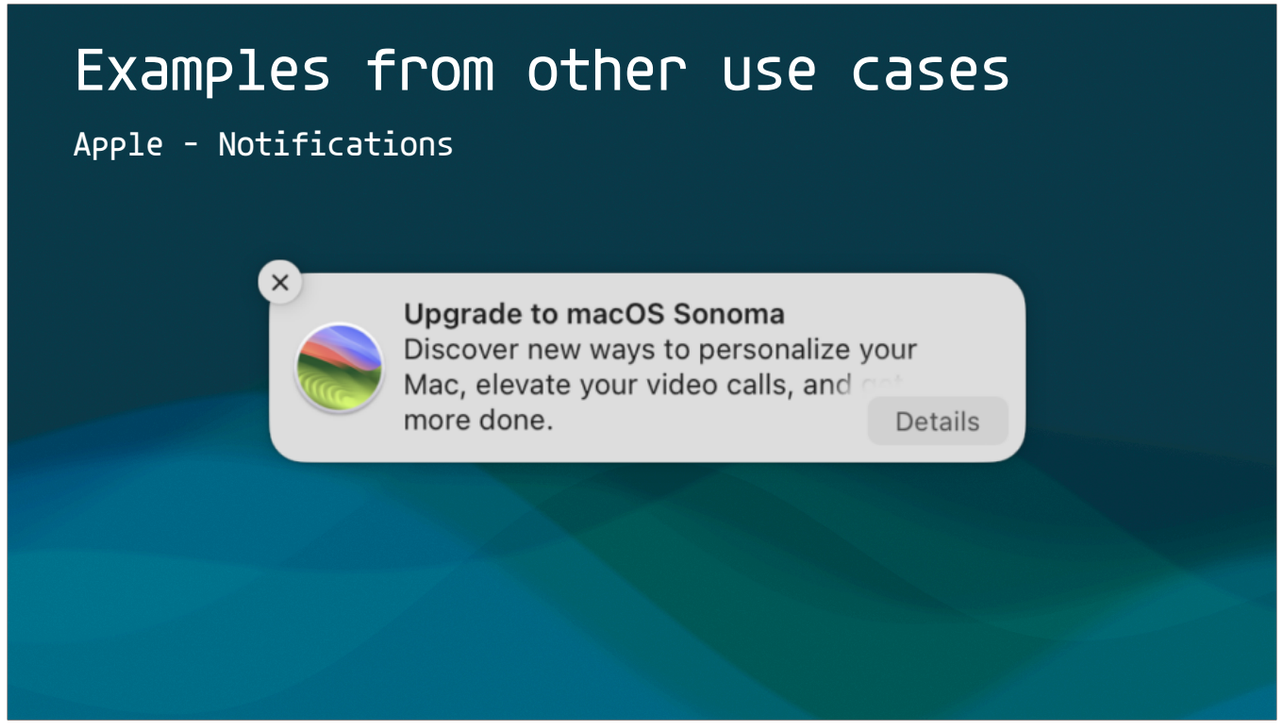


- Email
- Internal notice board/dev forum
- A wiki page
- Slack notifications
- ....repeated Slack notifications?

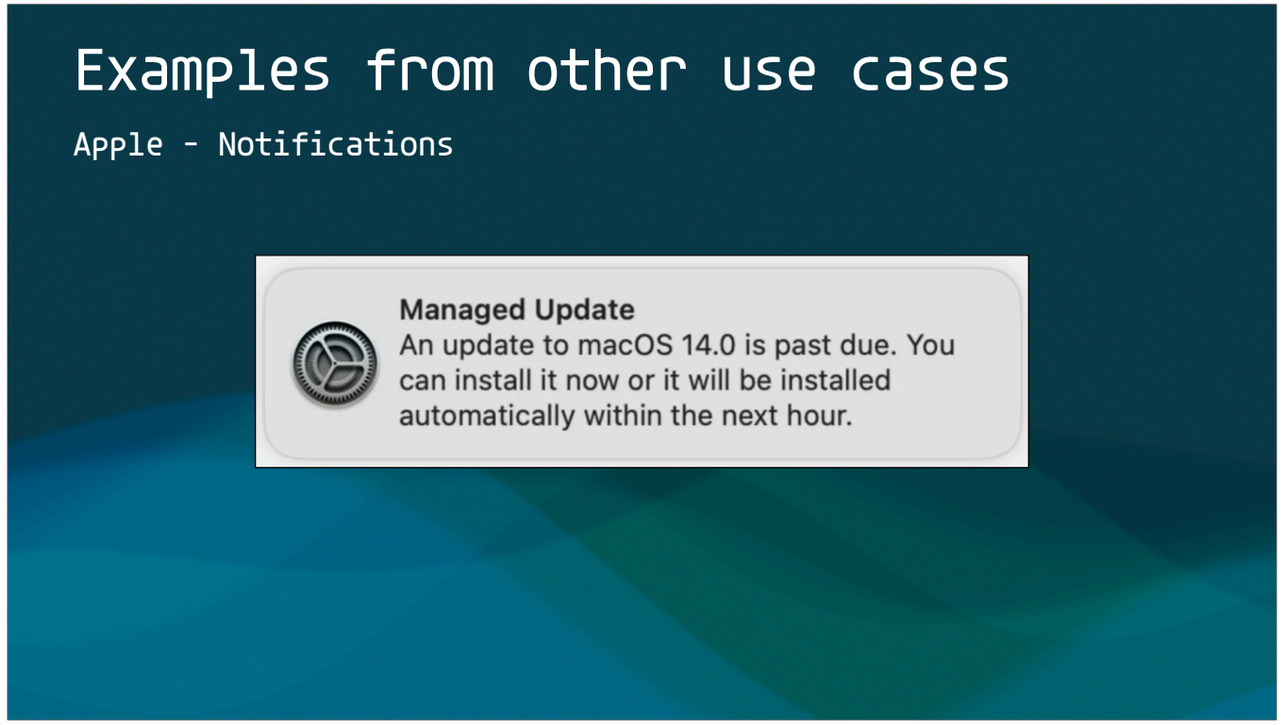
But will they read them? Maybe some, but most..NOPE

<CLICK>
We all know folks will not read these despite your best intentions when they need it most.

So what's the solution? What have others done for similar needs?



macOS has notifications, but those can be hidden during a Focus mode, a common feature used during meetings and one users can control using customized Focuses. These can also be dismissed easily.

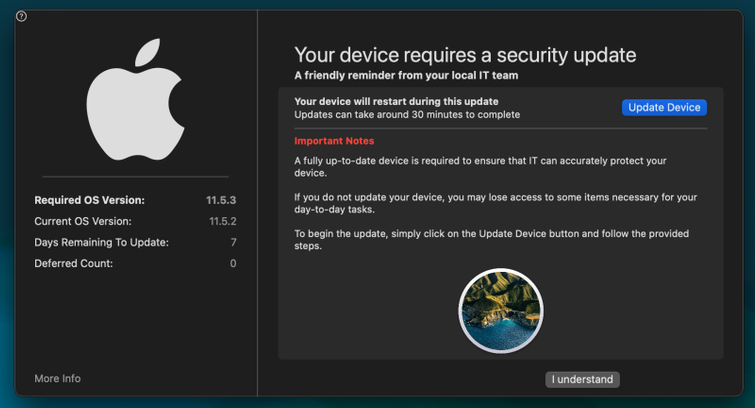


Thankfully for Software Updates, this is improving with Declarative Management. But this is for doing updates, not informing them of an important piece of info about the update.

Let's look at another example of prompting users to do their updates.

Examples from other use cases

Nudge



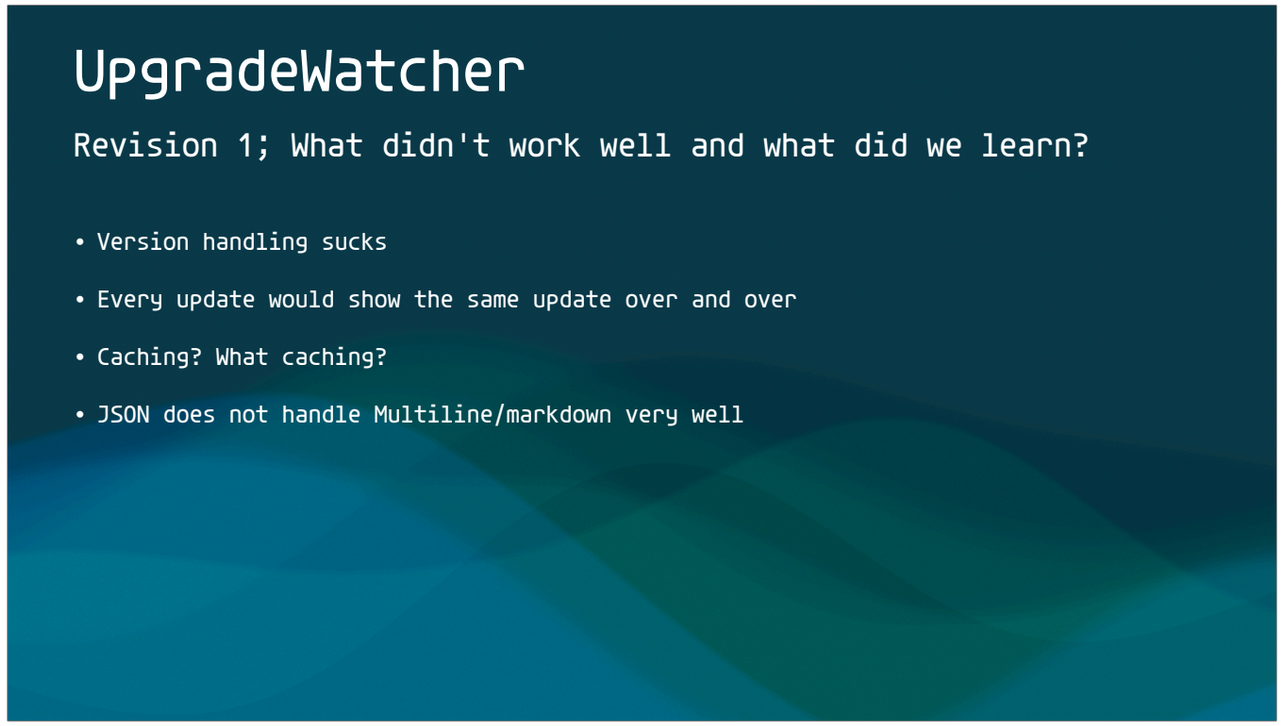
Huge thanks to Erik Gomez

https://github.com/macadmins/nudge/

Now this should be familiar to every one. But if not, this is Nudge. It simply alerts a user of a pending required update and gets out of their way. Many orgs have implemented Nudge or something similar for pushing users to the latest update to keep their devices secure.

This is a great model. It's there when it needs to be, but otherwise, it informs the user and gets out of the way.

And that's why I wrote...



When I first wrote it, it was quick and scrappy. There was an urgent timeline to get it done. Choices were made...some of them incorrect.
- Didn't handle versions well
- Every update would show the same update over and over
- Caching? What caching?
- JSON is not great for large multiline or markdown info

UpgradeBuddy

A much needed rewrite with much needed improvements

- Upgraded to SwiftDialog 2.3 allowing markdown documents to be used
- Switched to using Python's Packaging module for vastly improved version handling
- Added preference handling per user and tracking of seen vs unseen messages
- Caching of the message markdown
- Message format is now in YAML for clearer definitions **<- Don't @ me**
- A real schema to improve message creation

With time came SwiftDialog 2.3.0 adding Markdown document support.
- Switched to using Python's Packaging module for vastly improved version handling and specifier sets.
    - Why try and re-invent a massive set of wheels when there have been people before you that wrote it better.
    - https://packaging.pypa.io/en/stable/
    - This also allowed for better version of scoping for messages using specifierSets
- Added preference handling per user and tracking of seen vs unseen messages.
    - No more of the same message after every single update!
- Caching of markdown for messages, a better per message format using an actual message schema, etc.
- Messages is now in a YAML file instead of JSON
- A schema to improve message creation

UpgradeBuddy

How does it work?

- SwiftDialog for the dialogs - Thanks to @bartreardon
- Outset v4 for executing - Thanks to @chilcote @bartreardon
- S3 + CloudFront for delivery and global availability
- Optionally:
- CodePipeline to automate stages for testing new messages
- YAML documents separate each message

How does it work?
- SwiftDialog for the dialogs..obviously
- Outset v4 for executing on login
- Utilizing S3 + CloudFront for delivery and global availability to download the latest message document you've set in preferences
- Optionally:
- CodePipeline to automate stages for testing new messages
- Message document is written in YAML, Yes I know.
Let's see some messages we've used UB for.

UpgradeBuddy - Demo

CMDs in Obsidian

Where can we get it?

https://github.com/aws-samples/amazon-upgradebuddy-example

For now... :-)

SO where can you get it?

Why is it under samples, well for now... I'll answer that with the following image.

Why aws-samples....

Questions, Comments?

If None, let's go to lunch!