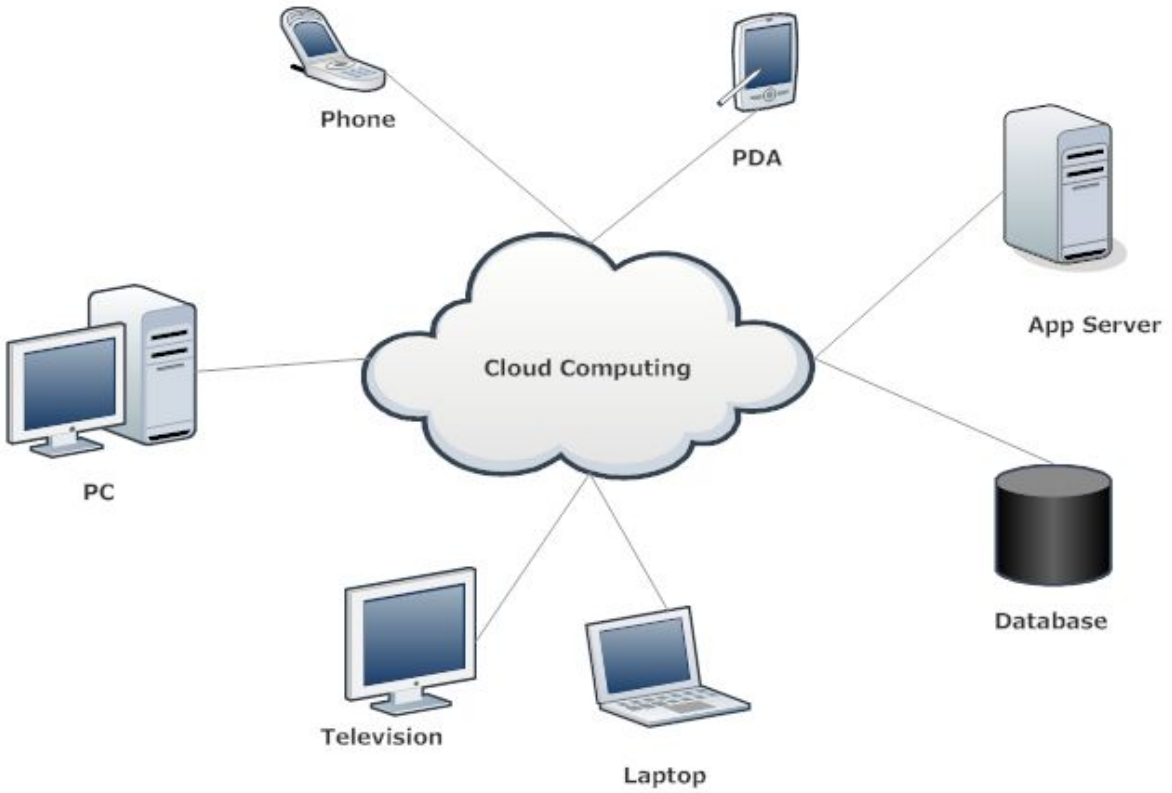
Web Application Development on the Cloud

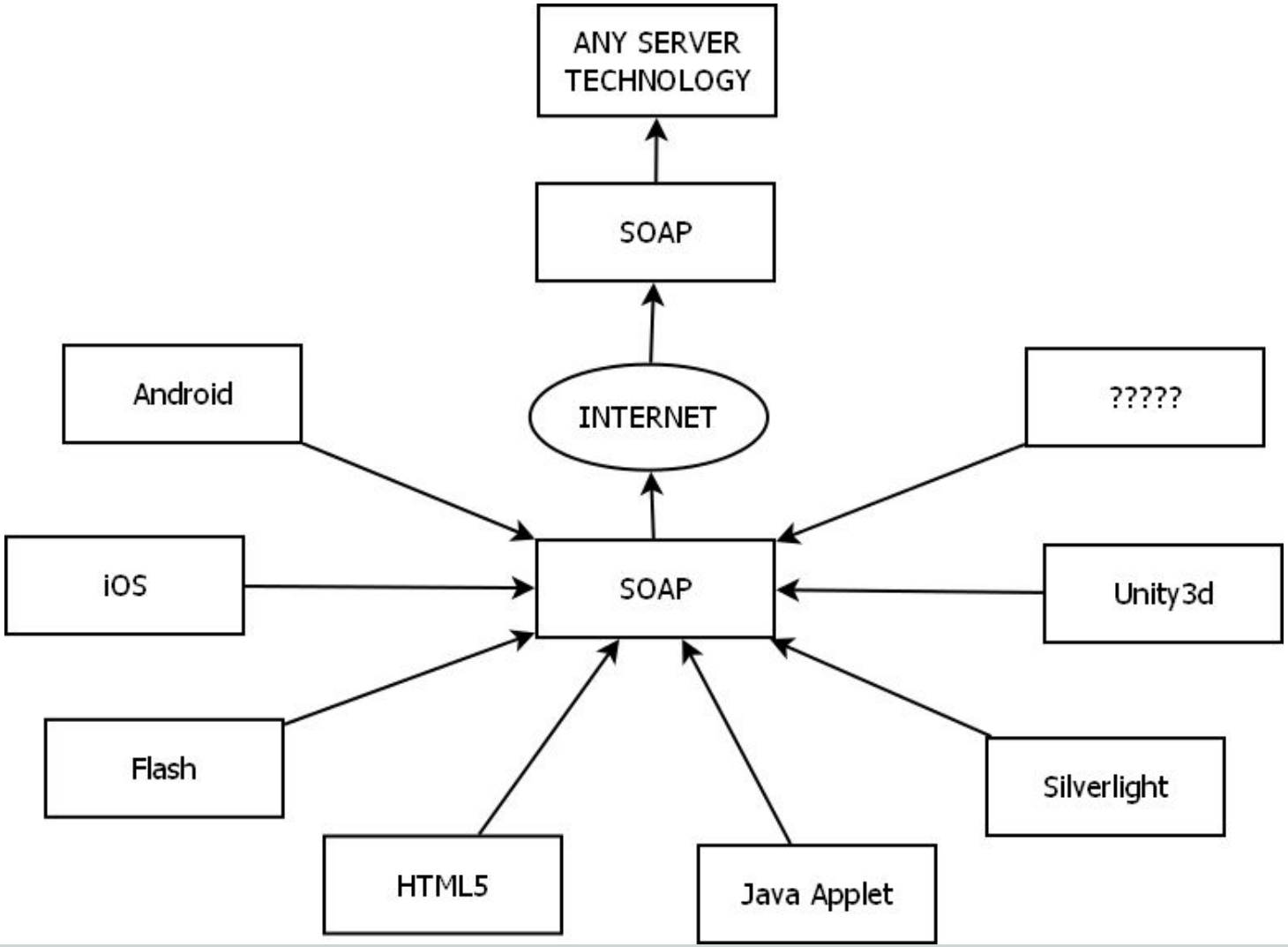
Platform-as-a-Service (PaaS)

This is developing for the web...

Heterogeneous Internet



How do we choose?



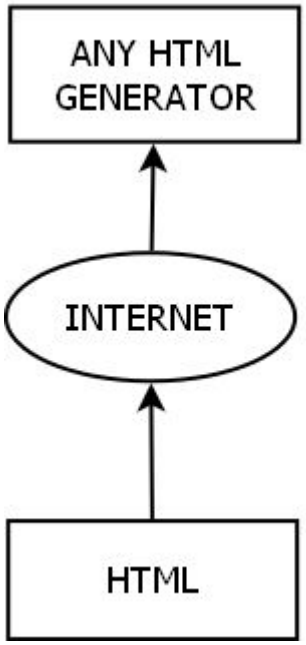
Keep technologies to a minimal

- Reduce dependencies
- Reduce development context switching
- Reduce time needed to keep up with updates to each technology
- Reduce redundant coding
- Reduce technologies to interface between technologies (BlazeDS, AMF to Java)

Web Browsers

Problems with Web Browsers

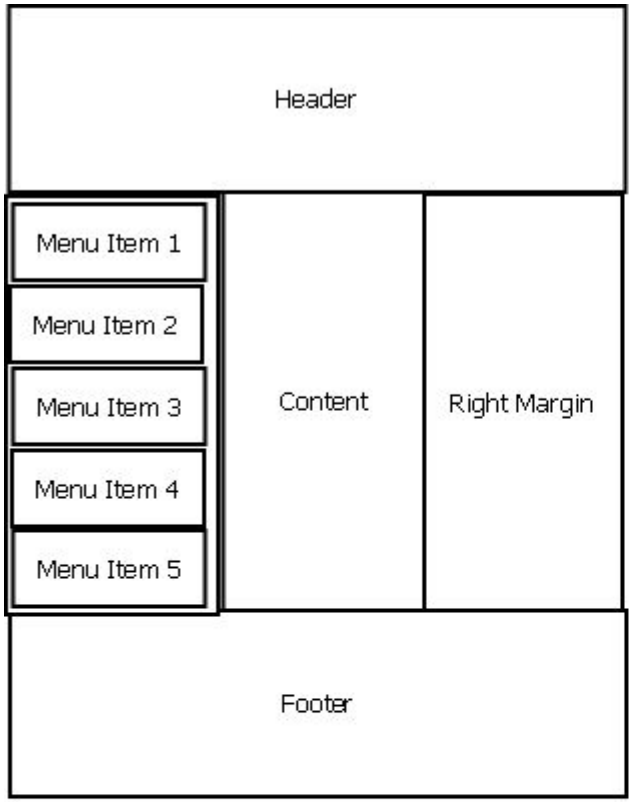
- Made for text, not applications
- Each browser is like its own OS, small differences in API
- Static set of UI controls
- Quickly becomes “hack fest” to get around artificial limitations (GWT, ASP.NET, AJAX)



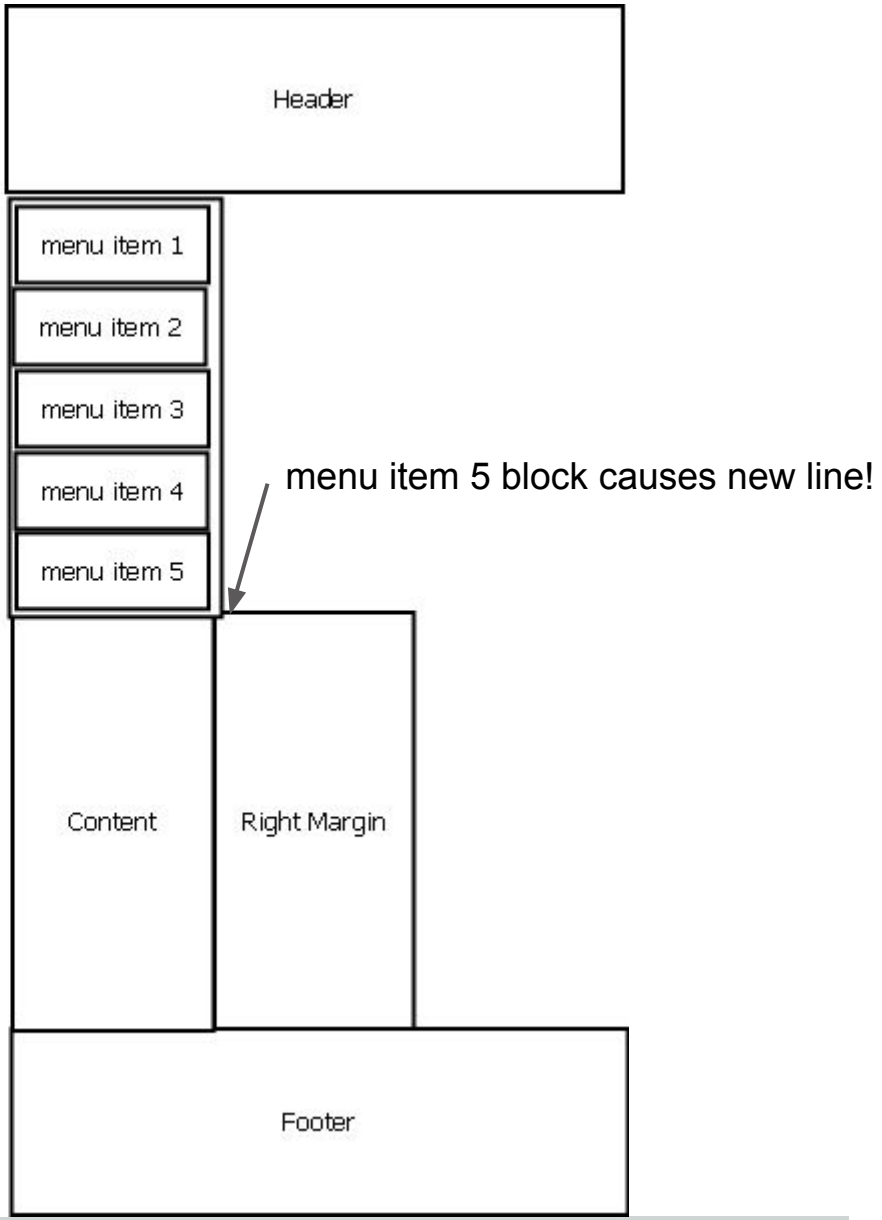
Solution: Plug ins!

- SUN Java Applet
- Macromedia Shockwave
- Adobe Flash
- Microsoft Silverlight
- Unity Technologies Unity3d

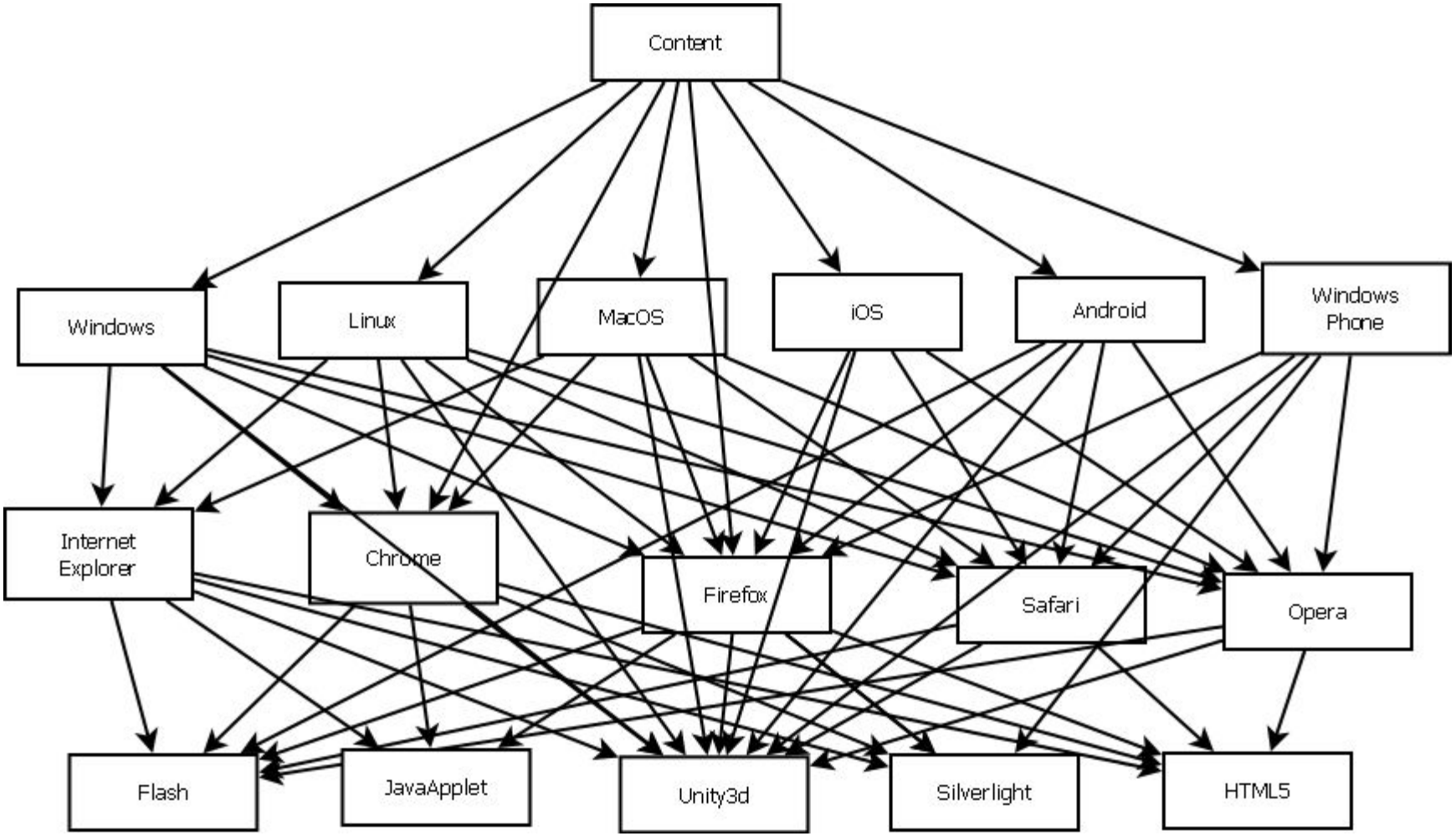
what we want



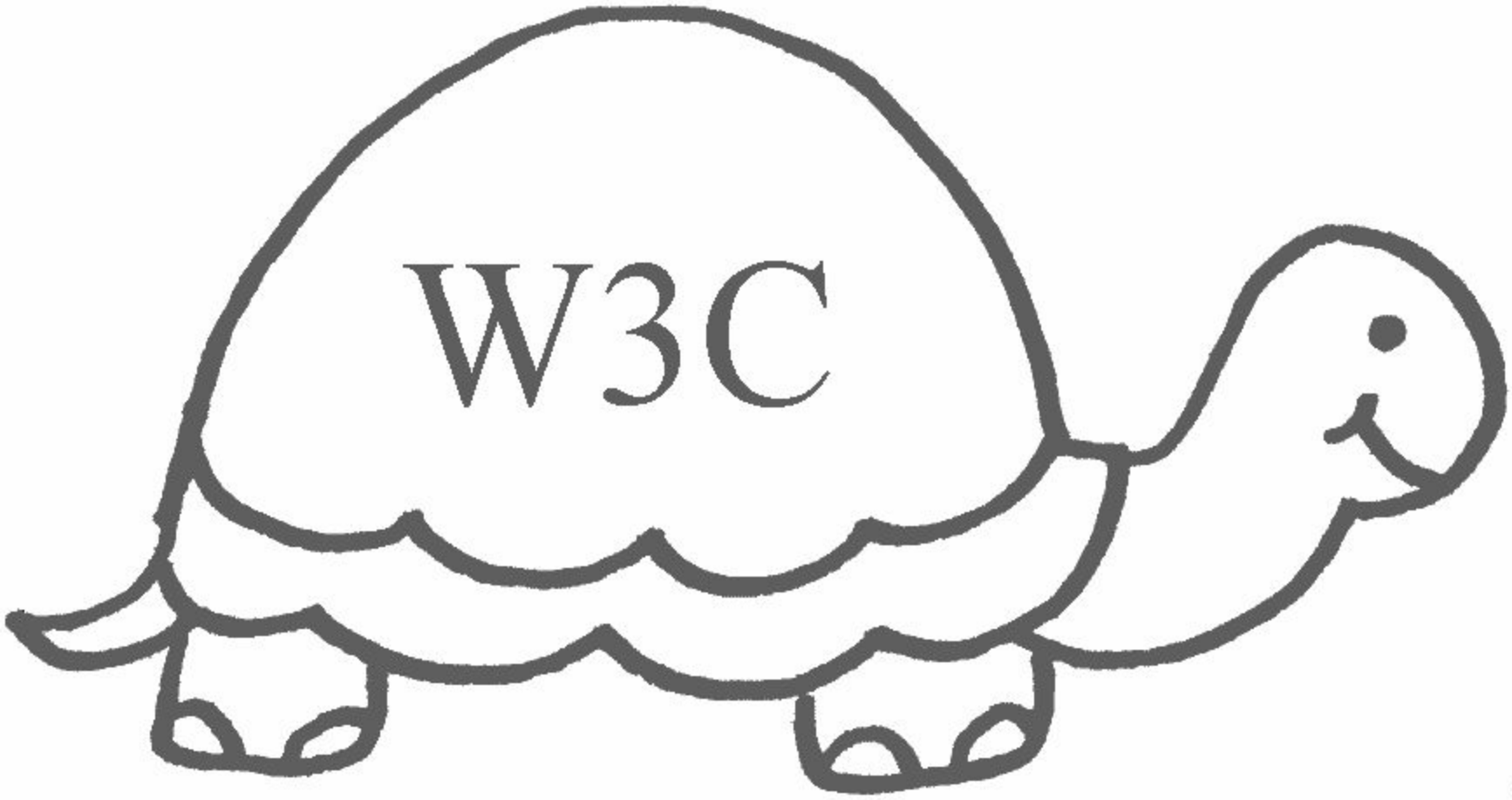
what we get



Mobile Devices



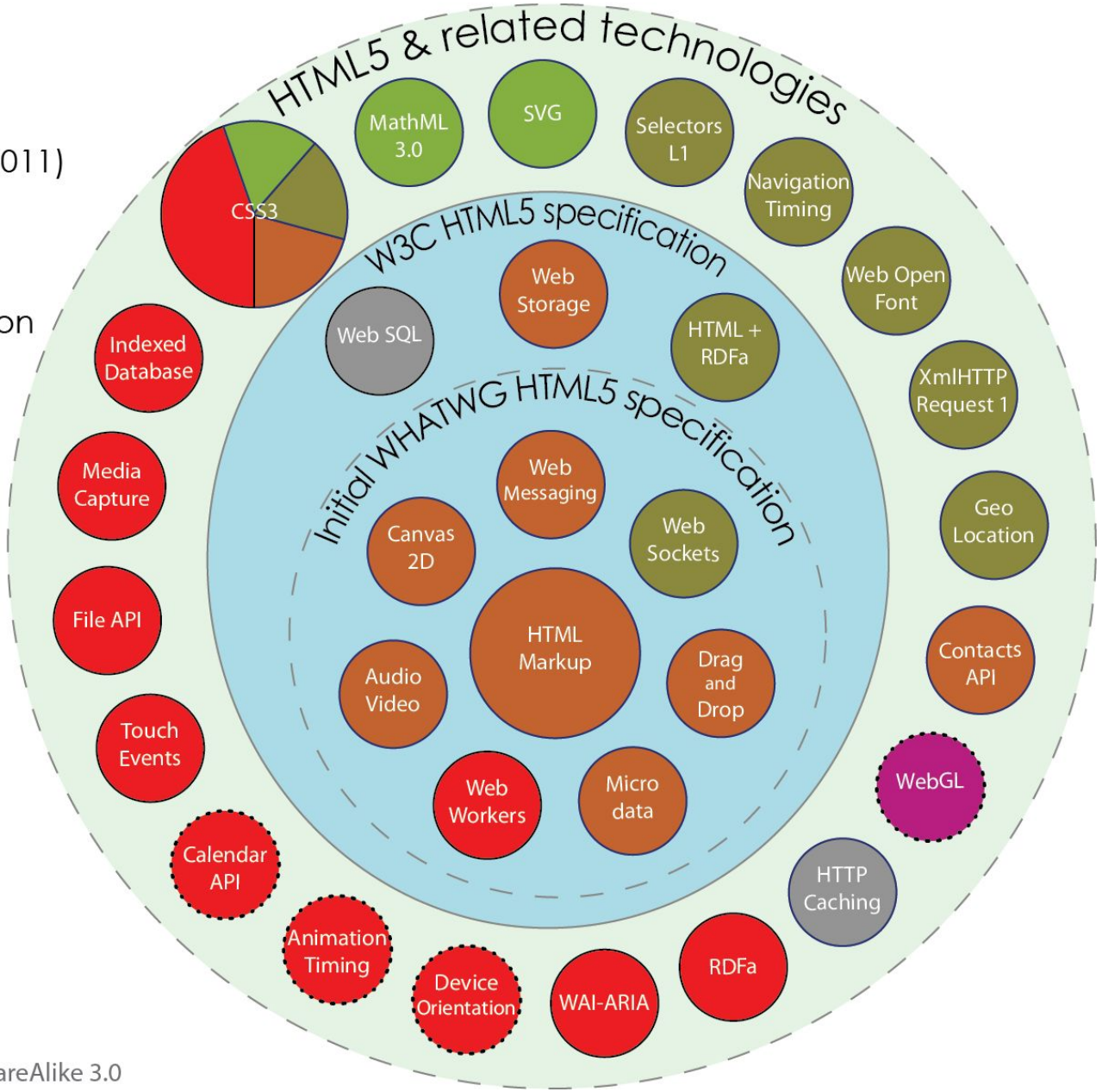

HTML


W3C

HTML5

Taxonomy & Status (December 2011)

- W3C Recommendation
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated W3C APIs



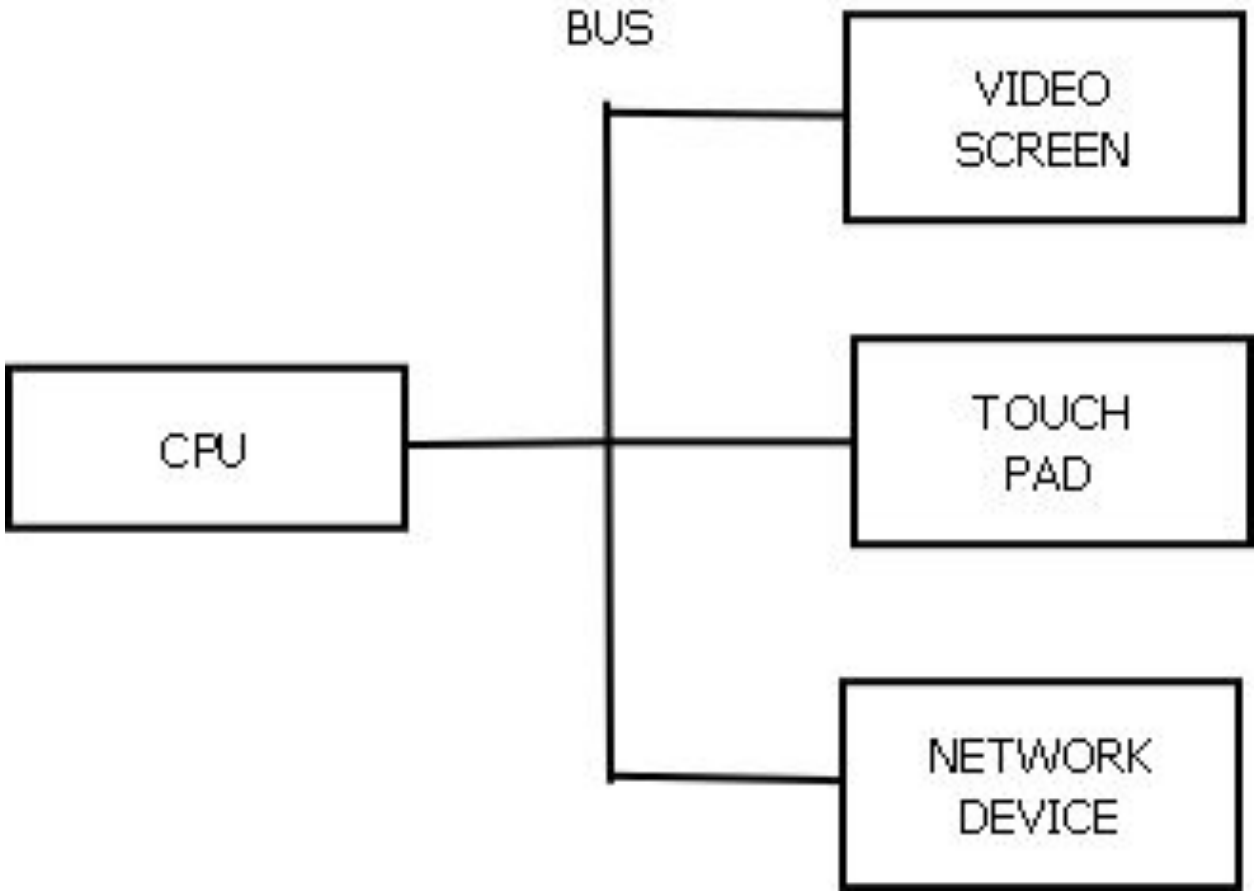
What are we trying to solve?

What is it about HTML5 that everyone is so excited about?

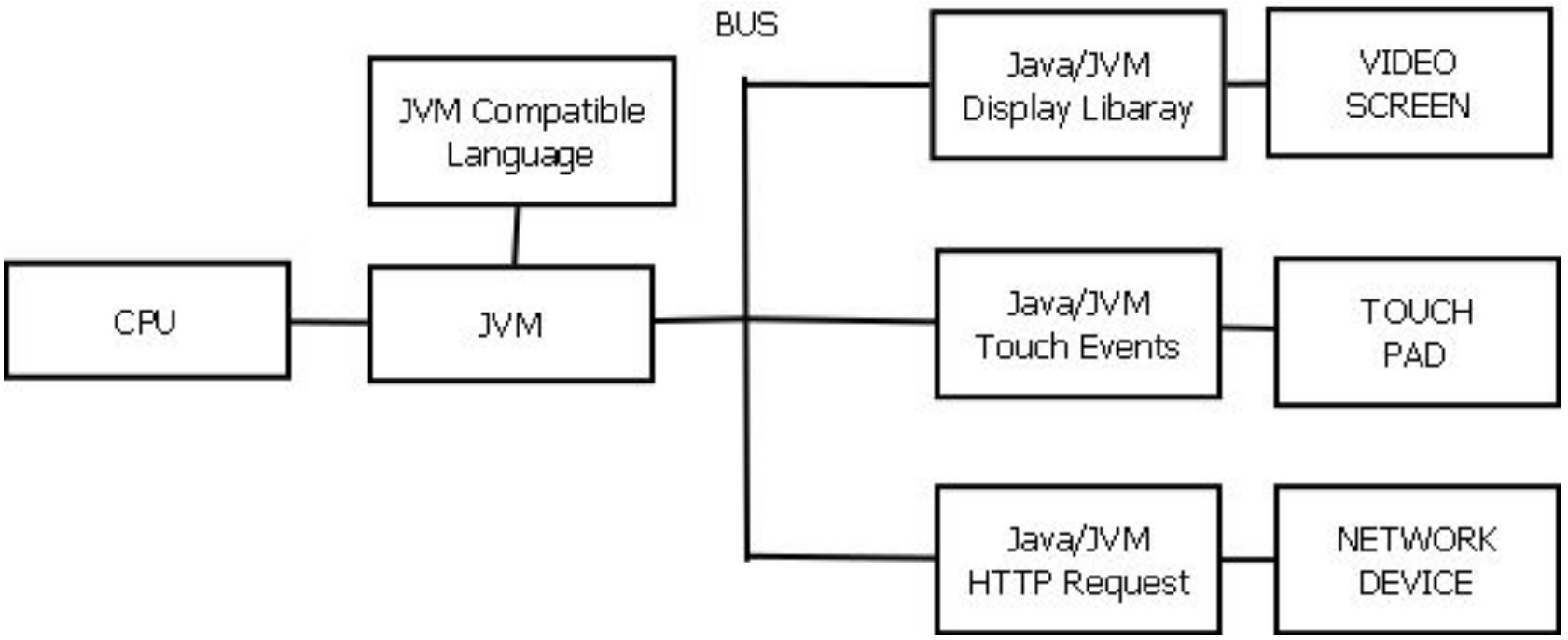
What common problem do all of these technologies solve?

Portability

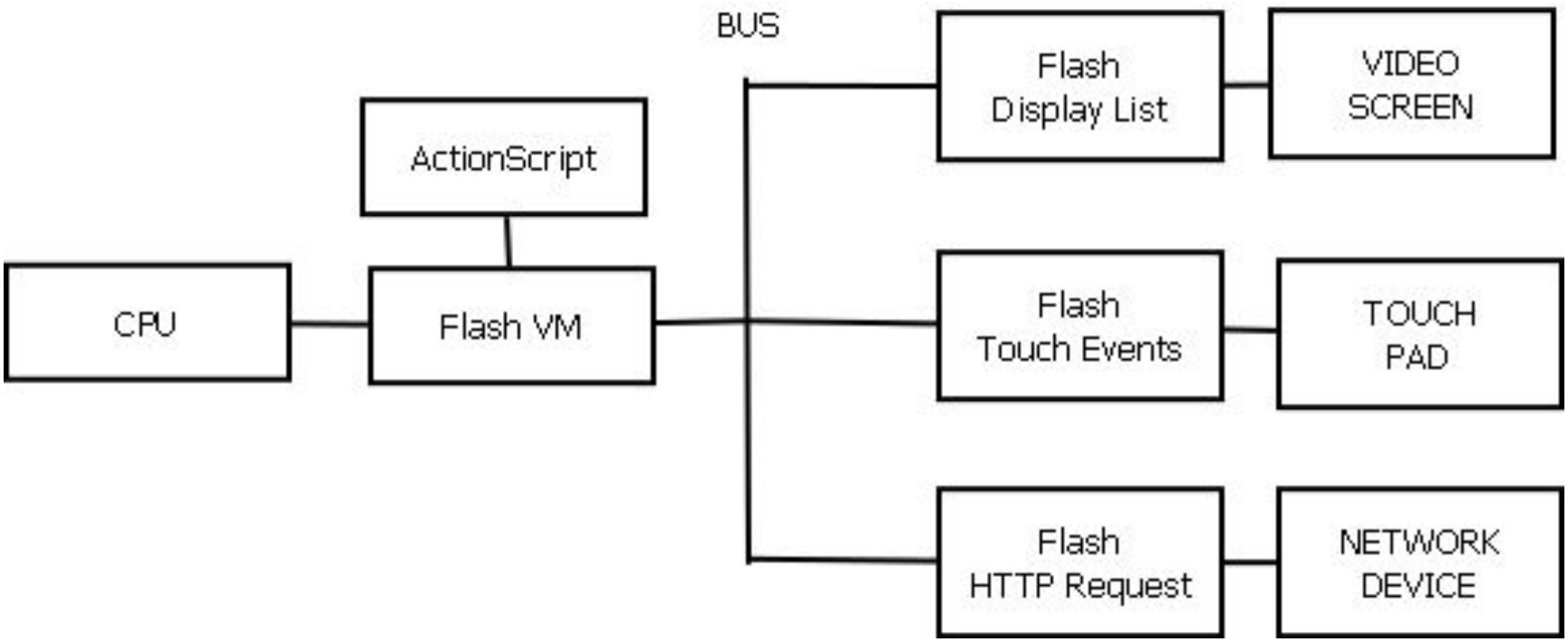
Virtually every device shares this structure



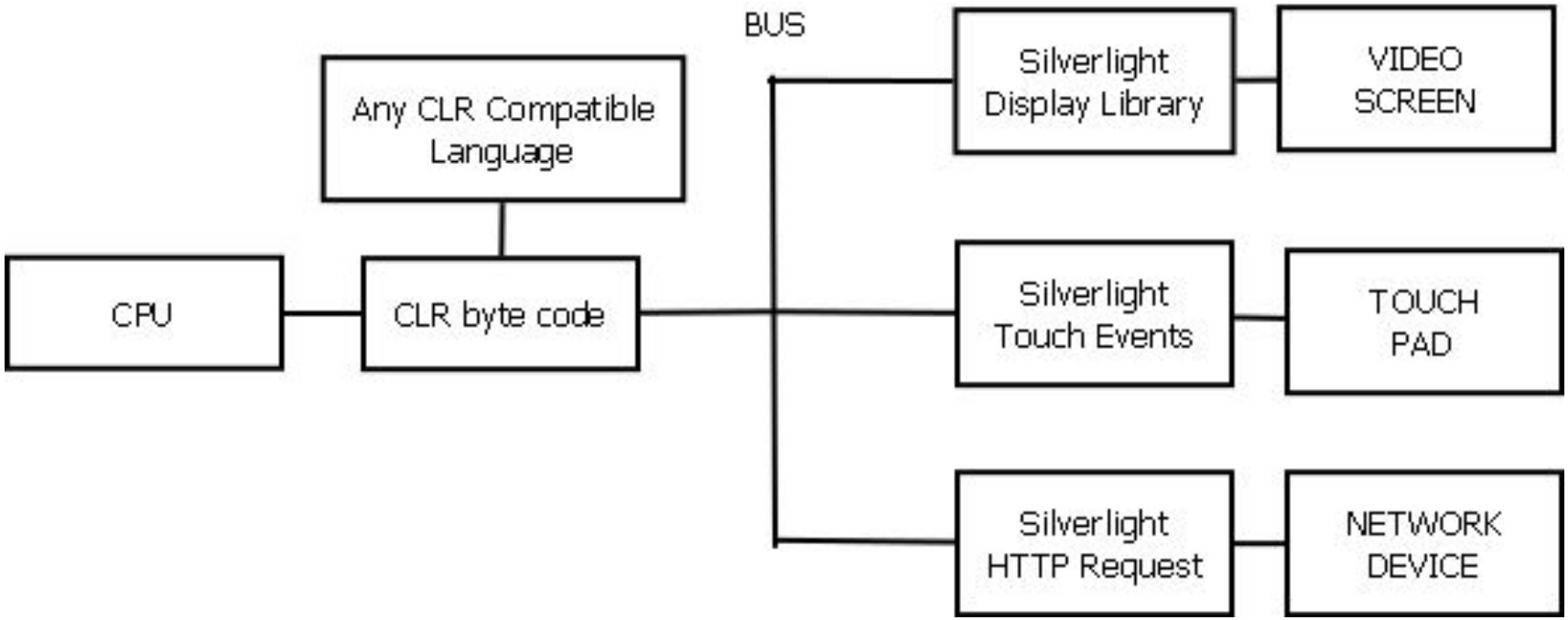
First there was Java



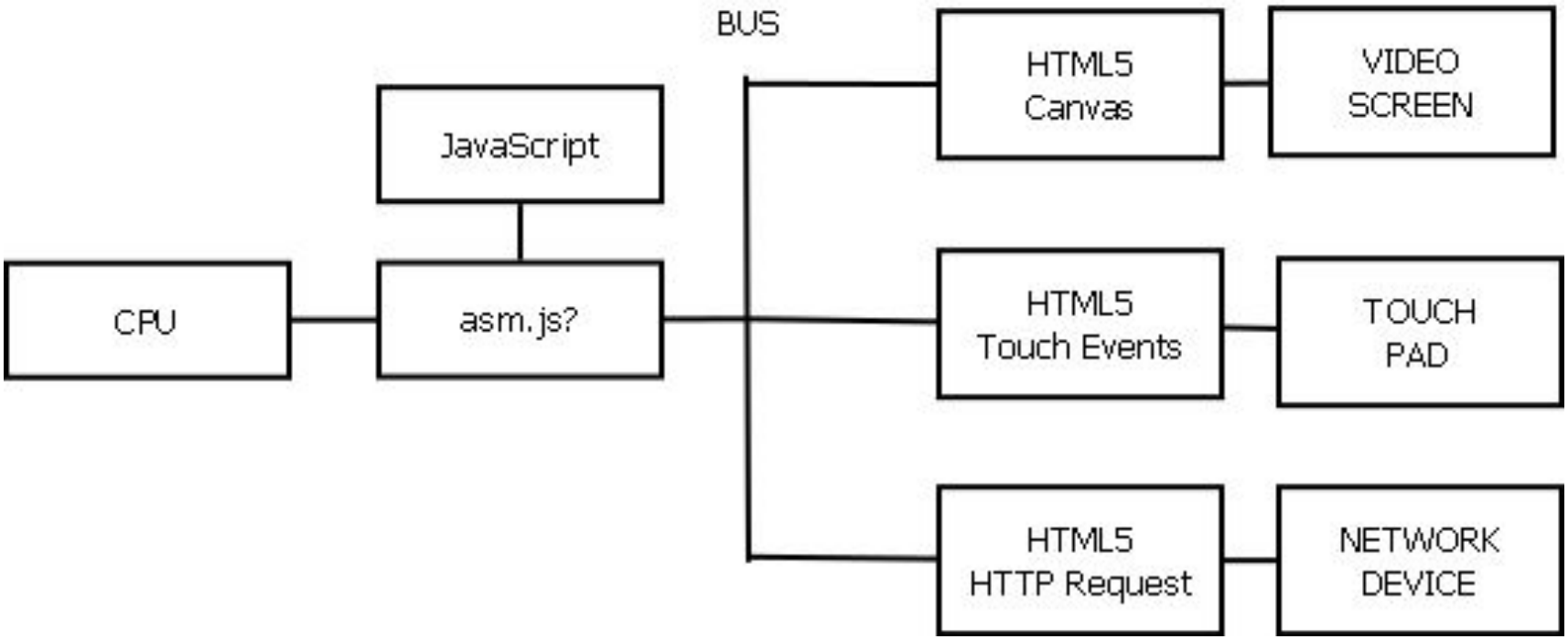
Then came Flash Player



And Microsoft with .NET, Silverlight



And now HTML5



It’s not the portability that people are excited about, it is the standardization

Some Limitations

- Slow to standardize
- Standards high level & numerous
- Development similar to Waterfall Model
- Widgets not as simple to extend
- Relies on slow JavaScript

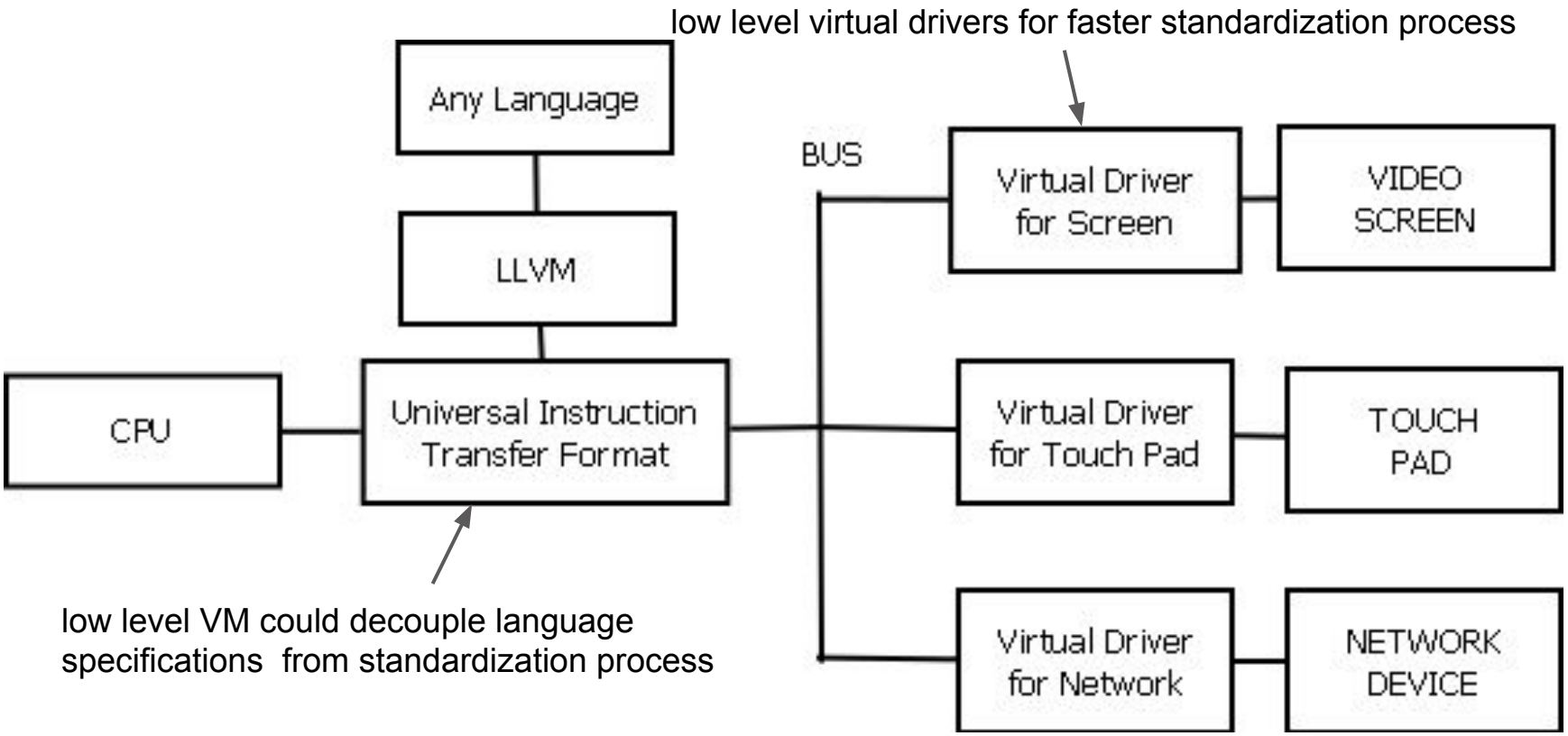
Dependency on JavaScript

- Nothing particularly interesting about JS
- Find less errors at compile time (GWT)
- Step backwards for language agnosticism
- Not designed well for parallelism
- There is a shift in how we think about code
- Game industry using data oriented design
- Mobile devices will get smaller, needing more efficient computations with less computing power

Why not have a Universal Instruction Transfer Format?

- UTF lets different OSes transfer files
- Why not let a UITF transfer binaries
- Chrome is already doing this using LLVM as a transfer format with NaCl
- Flash uses FlasCC to compile C++ code to LLVM, then convert to FlashVM byte code
- There is a project to compile C++ to Asm.js also using LLVM
- Flash also has AGAL, a transfer format for GPU shader programs in stage3d

HTML5 Becoming Universal Computing System



Browser acts as Virtual Operating System, managing processes and modular virtual device drivers

Some Server Considerations

- NoSQL databases
- Allowing Server Scripts to use RAM
- Cloud Operating Systems
- Concurrency/Threading
- Parallelism

Some interesting tech combinations

Silverlight => SOAP => ASP => DBO => Database

Java Applet => SOAP => JSP => JDO => Database

Flash => AMF => ColdFusion => dbo => Database

HTML5 => JSON => Node.js => dbo => Database

NaCl => XML => C++ => dbo => Database

HAXE/Flash => transfer => HAXE/Php => Database

Case Study

Flash => JSON => Php => Custom Php dbo => MySQL

References

See my “Application Development on the Cloud” paper for list of references