AJAX

Reality Check

Björn Müller, CaptainCasa GmbH

User Categories

- Quality of Usage
  - Tactical User ... Doer
- Frequence of Usage
  - Sometimes ... Frequent User
- Type of Data Processing
  - View ... Edit
- Relation to User
  - Anonymous ... Known

„Typical Power User“

- Quality — Doer!
- Frequence — Frequent!
- Processing — Display / Edit
- Relation — Known

- Examples
  - Call Center Employee
  - Purchasing Manager
  - Financial Accountant
  - Production Planner
  - Developer - IDE

Power User's Expectations

- User Interface needs to be...

- FAST
  - Response times to be measured in milliseconds
- ROBUST
  - Whole day usage without restart / slow down
- INTERACTIVE
  - Up to date control processing: drag & drop, popup menus, table sizing, ...
  - Keybooard-able
- SMART
  - Look and feel

„Typical Casual User“

- Quality — Tactical User ... Doer
- Frequence — Sometimes
- Processing — Display ... Edit
- Relation — Unknown

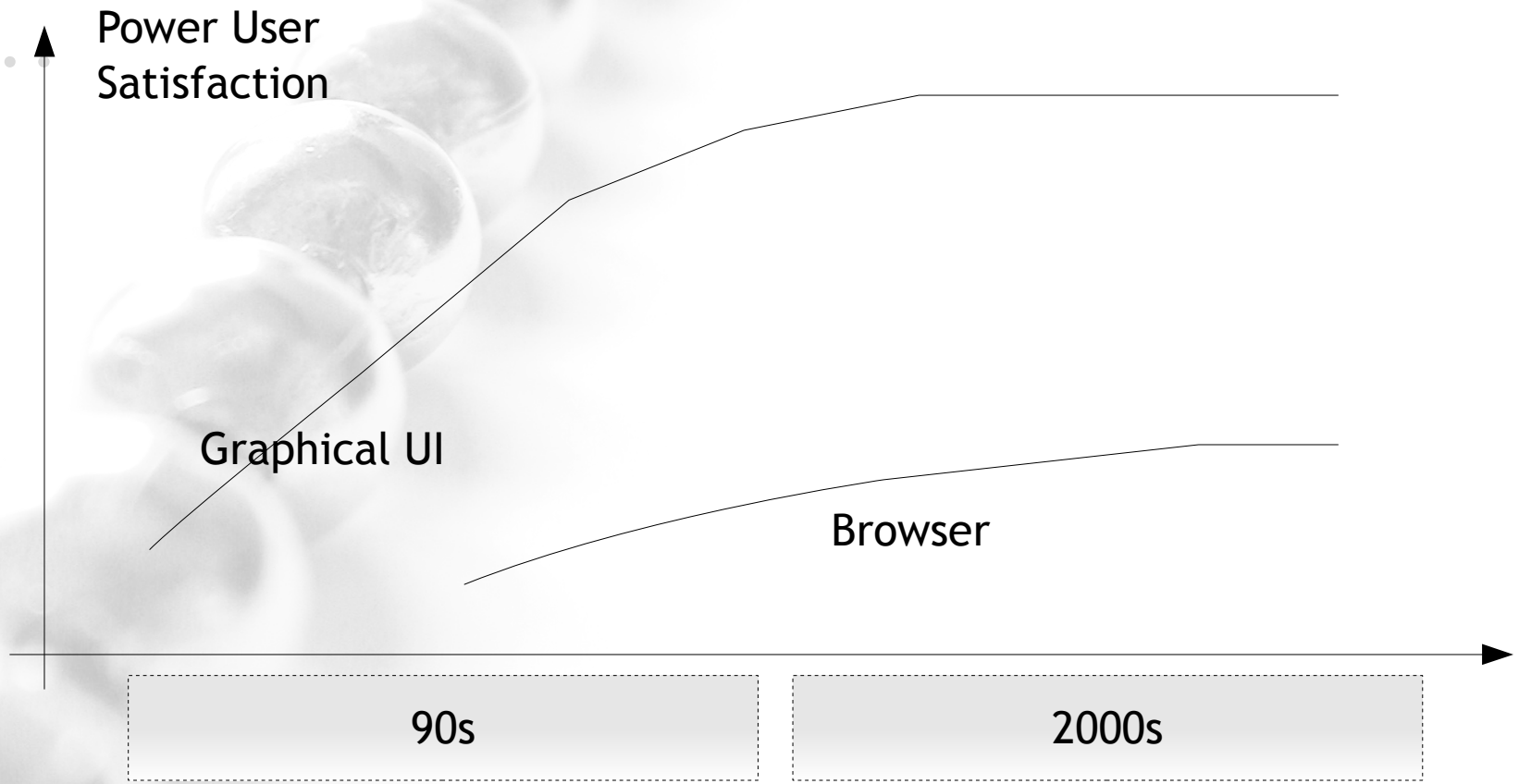
- Examples
  - Web Mail
  - Mini Office User
  - Travel booking
  - Developer - Newsgroup

Casual User's Expectations

- User Interface needs to be...

- AVAILABLE
  - No client installation / configuration
- SMART
  - Look and feel
- INTERACTIVE
  - Avoid „annoyance“
  - Example: calendar input, ...

No Hype around Power Users...



IT Manager's Expectations

- User Interface needs to be...

- SIMPLE TO INSTALL
  - Cost of Ownership
  - Cost of Maintenance
- EFFICIENT
  - Network resources
- ADAPTABLE TO APPLICATIONS
  - Fit to server side application processing
  - Software as a Service
- BASED ON STANDARDS

Developer's Expectations (I)

- User Interface needs to be...
- SIMPLE and EFFICIENT
  - Concentration on business content rather than concentration on technology research
  - Take over ugly tasks
- ESCALATION-ABLE
  - Profiling
  - Debugging
- SMART
  - Look & Feel
- BASED ON STANDARDS

Developer's Expectations (II)

- And...:
- Reduced effort for providing user interfaces for different user groups.
- „One UI should fit all!“ is an essential reason for the AJAX hype.

And here comes... AJAX!

- PREJUDICES (page one)
  - AJAX improves the interaction comfort
  - AJAX gives HTML pages a flavour of desktop applications
  - AJAX provides frameworks which manage the complexity of HTML / Javascript / ... processing
  - AJAX is (still) an area of hype, there is a lot of development going on worldwide
  - AJAX is complex, but there are frameworks that hide complexity
- GOOD NEWS: THEY ALL ARE TRUE!

And here comes... AJAX!

- PREJUDICES (page two)
  - AJAX client rendering is slow
  - AJAX frameworks either work fine with IE or with Mozilla
  - Using AJAX frameworks means you should know the private phone number of the framework developers
  - Browsers are oracles
- BAD NEWS: THEY ALL ARE TRUE. Sorry.

AJAX Performance Challenges (I)

- JavaScript X * slower than Java
  - No pointers, everything is an array/hashtable
  - Everything is interpreted, no JIT compiler
  - Very open to bad performance programming
- DOM Operations ...are expensive!
  - document.getElementById()...
- Nightmare: Javascript virus checkers

AJAX Performance Challenges (II)

- Size of base library that will be loaded with first screen
  - >, >> 100 kiloBytes
  - Loading can be buffered, parsing cannot be buffered
  - „Our framework only requires some kilobytes of JS code on client side to run.“ - Never believe this!
- Browser providers (Microsoft, Mozilla) do not seem to make substantial progress in the area of performance.

AJAX Performance Challenges (III)

- Browsers like rendering full pages. That's what they are originally designed for.
- Browsers do not like changes within existing pages.
- Best example: table rendering

AJAX Performance – Developer's Mistake

- When using an AJAX framework developers often forget about the performance impact of AJAX
- „It looks like native components, so let's build screens we know from native environments.“
  - Many controls on one screen
  - Many dynamics
- Developers need to be aware of working in a limited environment!

AJAX is still waiting for the JIT Effect

- ...is it likely to come?
  - No visible investments in JavaScript / DOM performance both for Internet Explorer and for Mozilla
  - Apple Safari pushes performance issues, but is not (yet) present in the area of enterprise usage
- Microsoft pushes Silverlight

- Will the JIT effect ever happen...?

- Waiting for better client hardware is a critical option.

AJAX Cross Browser Challenges

- Relevant browser platforms
  - Microsoft IE
  - Mozilla, Firefox etc.
  - Safari (?)
- Huge effort within AJAX framework development is spent on cross browser issues
  - Eventing issues
  - Sizing issues (height=100%)
  - Performance issues

Framework Abstractions

- Many frameworks are available that try to hide AJAX complexity by providing abstraction layers
  - JS libraries, OO component models for Javascript
  - Communication to backend
  - (Server side binding of application logic)
- Problem
  - UI components have many attributes
  + UI components can be flexibly arranged
  + Cross browser issues
  = Number of combinations is „infinite“
- Consequence: robustness of user interfaces is not adequate

Browsers are Oracles

- Very difficult to find out reasons for e.g. performance problems
  - Debugging
  - Profiling
- In addition: different behaviour dependent from browser
- Browser escalation management still is a nightmare compared to e.g. Java development

- Maturity level of browser environment still is not adequate

AJAX Maturity Conclusions

AJAX Maturity Conclusions

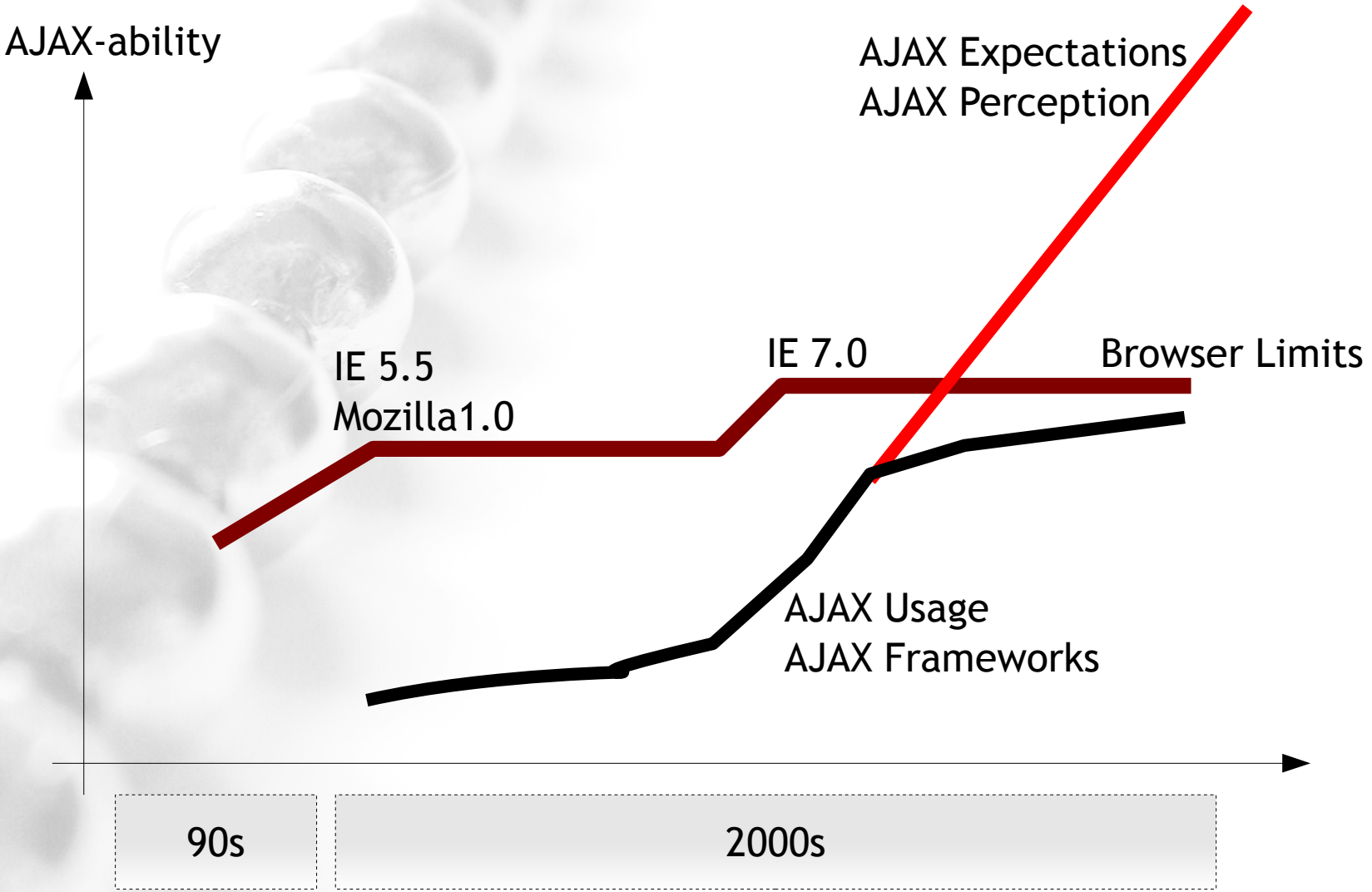
GoogleEarth

GoogleMaps

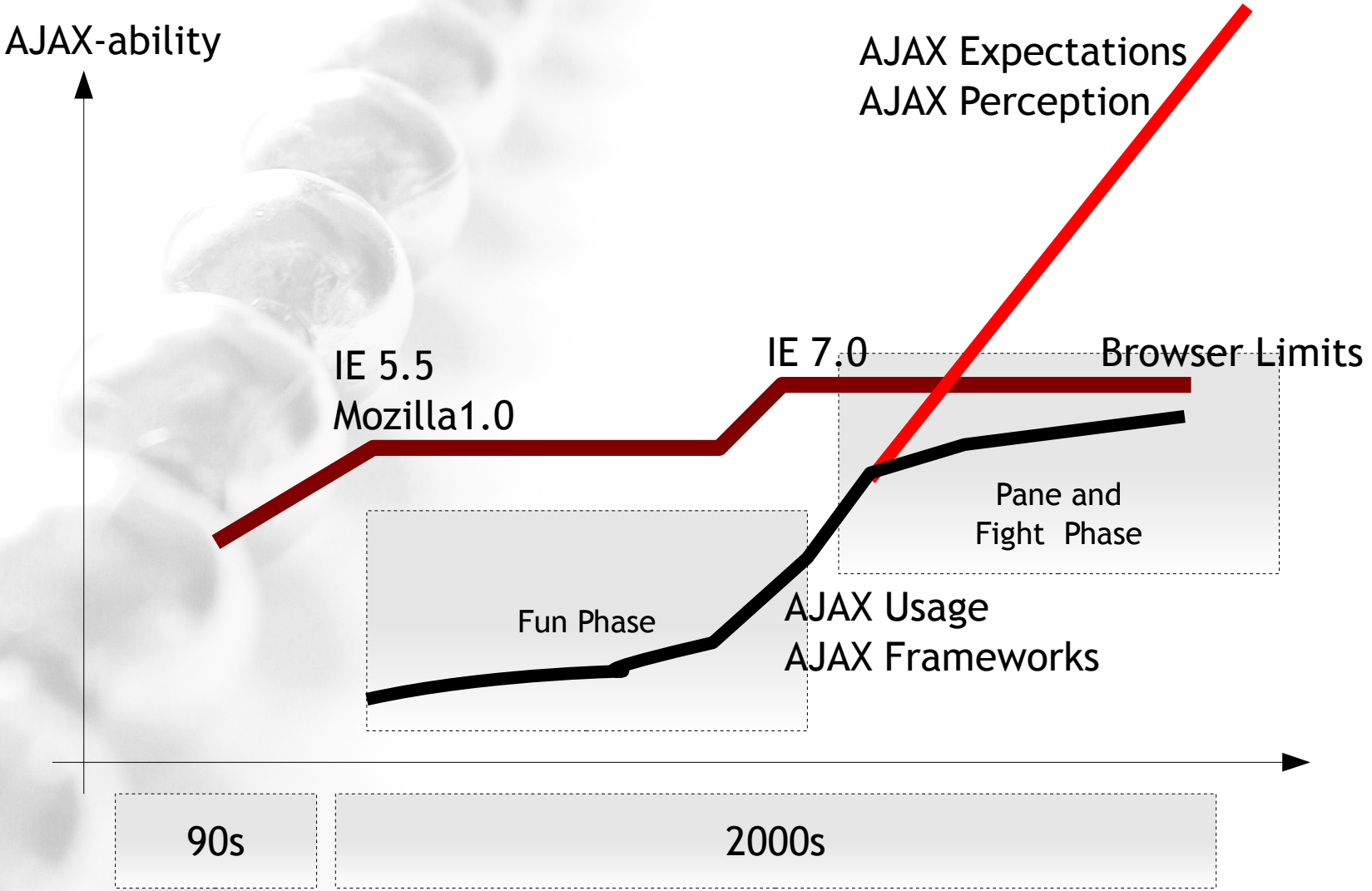
Online Demo

- AJAX based User Interface compared with Java Swing based User Interface

AJAX means „Fighting with Limits“



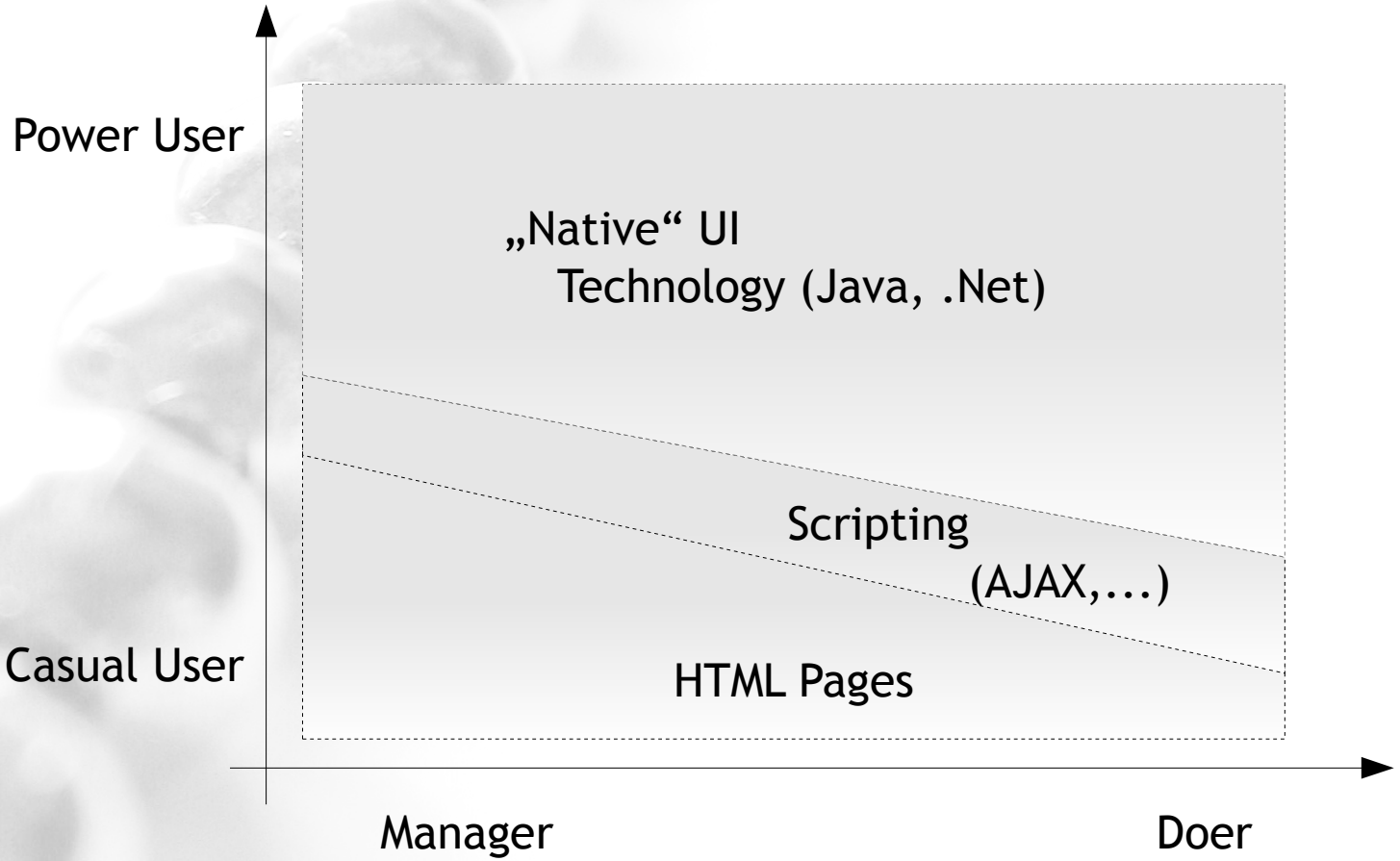
AJAX means „Fighting with Limits“



AJAX Usage Conclusions

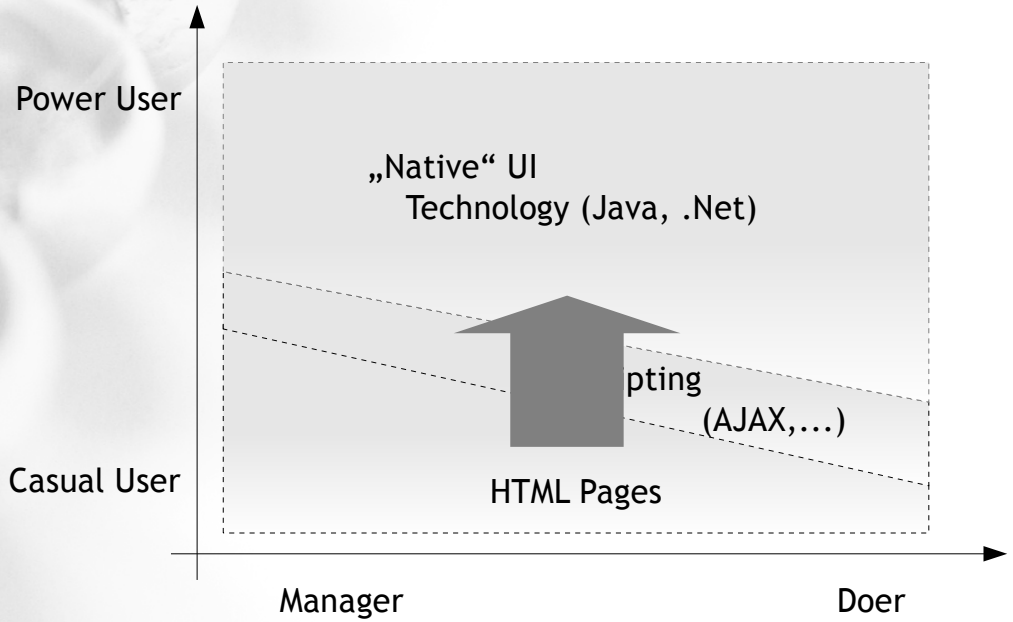
- AJAX based approaches in general are a risky approach for implementing UIs for operational users and power users
- AJAX is (very) usable in the area of applications...
  - Which are casually used
  - Which are „editing“ applications
  - Which are non mission critical in means of UI performance and UI client robustness
- Plain HTML is still the very best for „displaying“ applications.

User Types and UI Technology



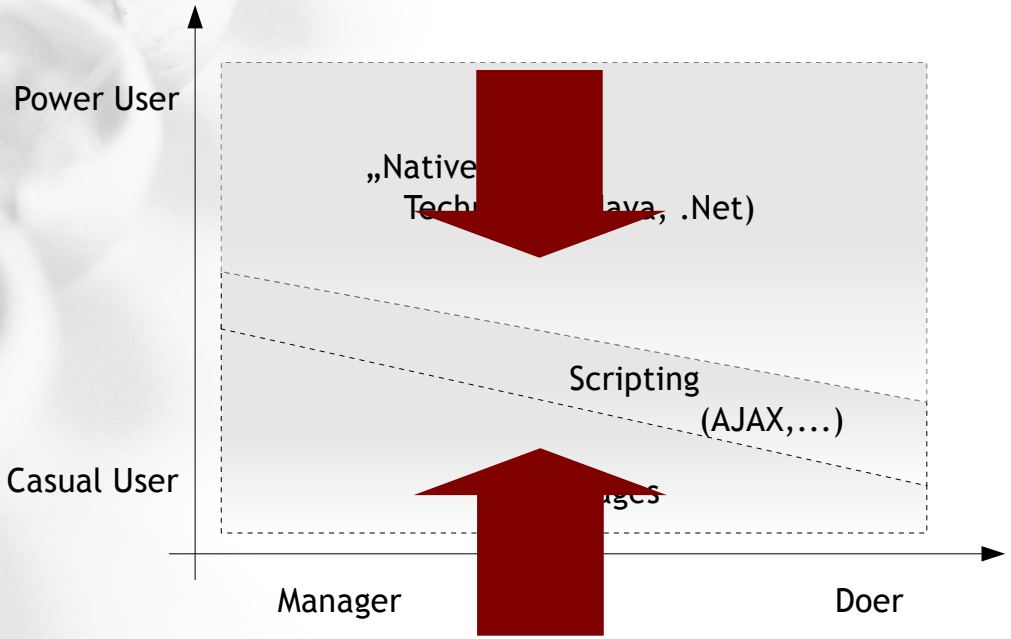
AJAX Expectations

- Lift up the are of HTML browser usage into the area of operational users.



AJAX Usage Dilemma

- Casual users want simple screens to be loaded fast!
- Operational users want complex screens to operate fast!

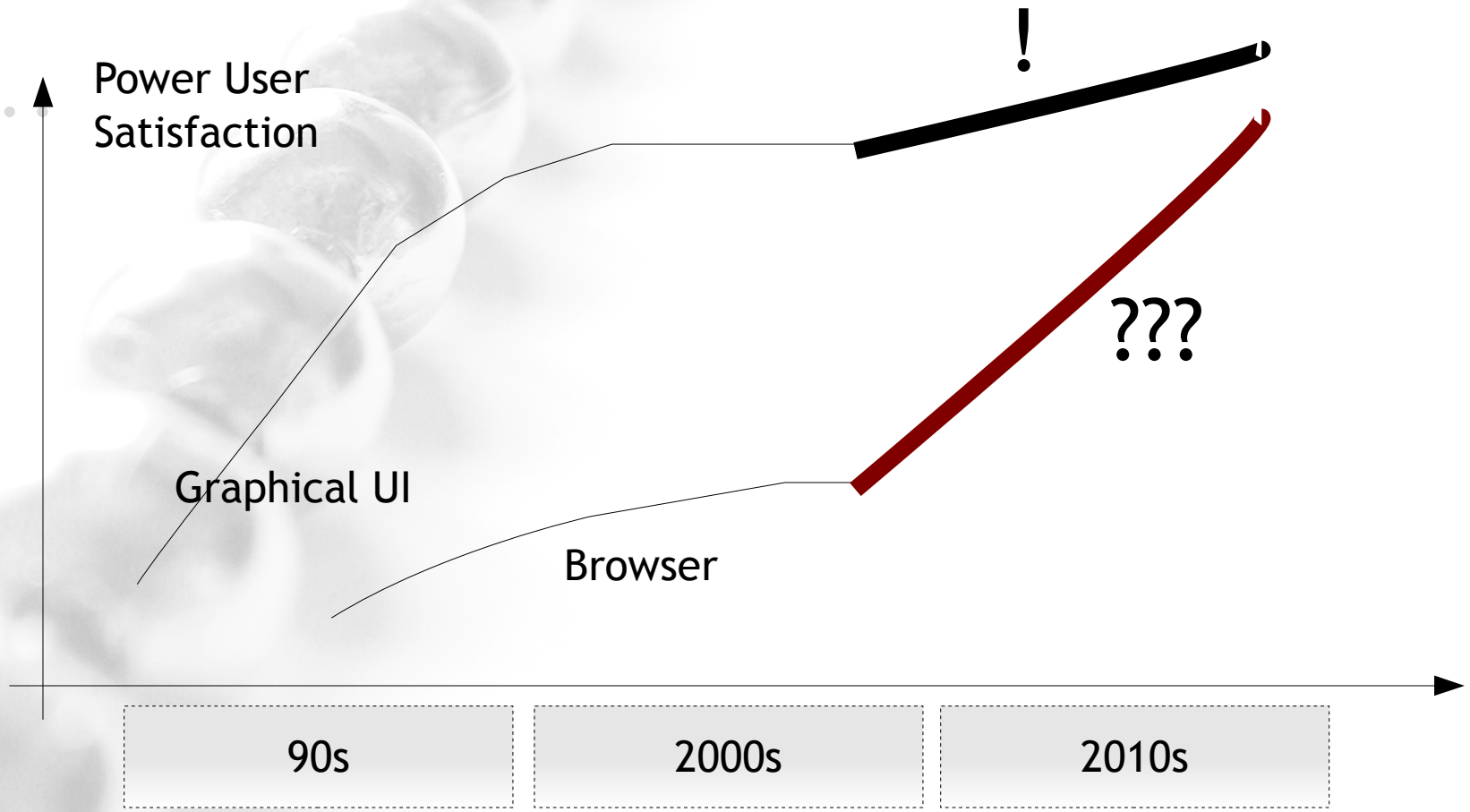


Challenge

- How do we serve our power users?

- ...and serve our server side enteprise applications
- ...and serve our install-ability issues
- ...and serve our „one UI fits all“ expectations

Comeback of „real UI Environments“



Comeback of „Real UI Environments“

- Java Swing / SWT Clients
- .Net Clients

- Difficult to categorize
  - Adobe Flex
  - Microsoft Silverlight

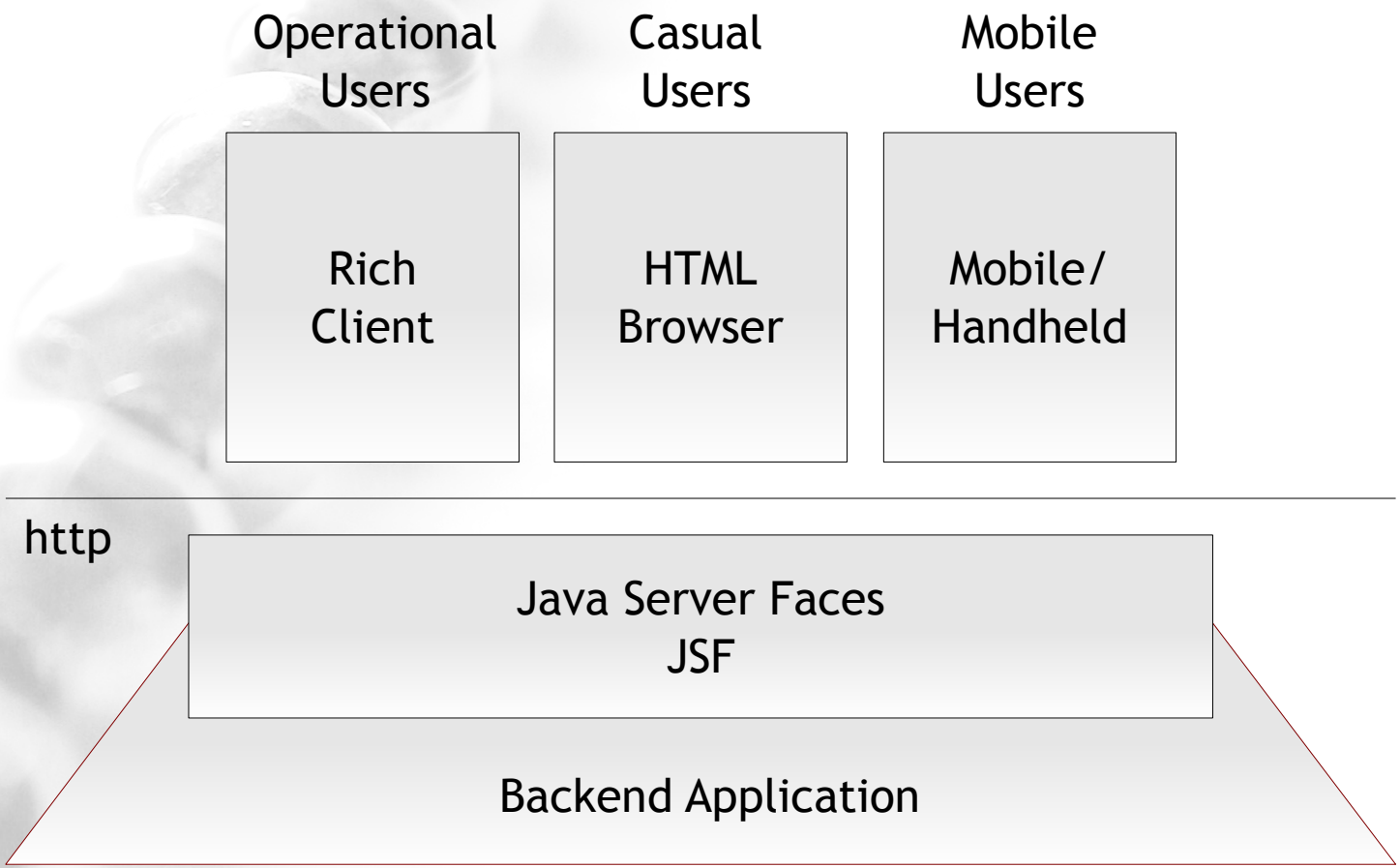
- Cost of ownership issues are discussed more open than some years ago
  - „Willingness“ to allow frontend plugin installations, as long as they are manageable

Comeback of „Real UI Environments“

- BUT:

- Usage of „Real UI Environments“ must be done with today's IT expectations in mind!
  - Server side applications (SaaS!)
  - Install-ability within the frontend
  - Cross OS run-ability

Java Server Faces



Conclusion

- Ways to build Power User Applications
  - „Read Usage“ as HTML as possible
  - „Edit Usage“ come back of „real UIs“ but with modern architectural background
- AJAX
  - Good for „edit usage“-applications for casual users
  - Treat AJAX and AJAX frameworks in a fair way...:
    - Know about limitations. Accept limitations.
    - Live inside this limits. Pay attention when transferring your „fat client expectations“ into AJAX...!
  - Keep on being patient, friendly listeners to the oracles.