If, when and how?

Strategies towards web-based tooling

Jonas Helming & Maximilian Koegel
jhelming@eclipsesource.com
mkoegel@eclipsesource.com
Disclaimer - strategic content ahead!
What is web-based, what is cloud-based?

=> Tool is deployed in the cloud and accessed via the browser
Why web-based?

1. Modern UI technology

2. Simplified deployment
Challenges with web-based technology

- **High uncertainty** about frameworks’…
  - maturity
  - mid-term maintenance
- **Smaller ecosystem** for tool components:
  - fewer components
  - lower level of abstraction
  - less features

=> **Higher cost** (order of magnitude)
Evaluate Use Cases against NFRs

Use Cases

Evaluation of the benefits

Use Case A

Use Case B

Use Case C

+ + + + $$$$  
+ + $  
+ - $$
Criteria to evaluate the benefits

NFRs:
- Installability
- Portability
- Performance and responsiveness
- Usability
- Cost
Installability
Installability

- **Pro-web use cases:**
  - Tutorial
  - Evaluation version (marketing)
  - Code review
  - Quick contributions (e.g. open source)

- **Contra-web use cases:**
  - Small steady development team
  - Using standard IDEs

- **Key decision criteria**
  - Complexity of installation (on a desktop)
  - Number of users
  - Expected period of usage

- **Alternative mitigations:**
  - Simplify set-up
  - Automated set-up (e.g. Oomph)
Portability
Portability

● **Pro-web Use cases:**
  ○ Tutorial
  ○ Read-only and review
  ○ External “non-PC” users
  ○ Web-Site / Content Management System
  ○ Complex and specific runtime setup

● **Contra-web Use cases:**
  ○ Highly unified and default hardware and runtime

● **Key decision criteria**
  ○ Sharable and specific runtimes?
  ○ Expected number of types of end user devices
  ○ Types of end user

● **Alternative mitigations:**
  ○ Virtualization of runtimes only
  ○ “Remote Desktop”
Performance and responsiveness
Performance

● Pro-web Use cases:
  ○ Very long compilation time
  ○ Extensive code analysis

● Contra-web Use cases:
  ○ 3D Modeling

● Key decision criteria
  ○ Extreme requirements on low-latency
  ○ High performance requirements

● Alternative mitigations:
  ○ Virtualization of performance-hungry operations only
Usability
Usability

- **Pro-web Use cases:**
  - Tools for web-developers

- **Contra-web Use cases:**
  - Tools for desktop-developers

- **Key decision criteria**
  - User Type
  - OS and hardware interactions

- **Alternative mitigations:**
  - Electron-based applications e.g.:
    - VS Code
    - Eclipse Theia
Cost
Cost

- **Pro-web Use cases:**
  - Slim code editors
  - Existing tooling is based on “Cobol 1.0”
- **Contra-web Use cases:**
  - Full-blown UML tool
- **Key decision criteria**
  - Which technologies are available?
  - Which developers are available?
  - What framework support is already there?
  - How much of the existing feature can be reused?
  - How many users do you expect?
  - Your available Budget
- **Alternative mitigations:**
  - Go for incremental MVPs
  - Pick the low hanging fruits first.
Short Summary on if
When?

- **Now**: Define a strategy and timeplan
- **Short-term**: Consider for architectural decisions
- **Mid-term**:  
  - Prepare architecture for migration
  - Migrate high-value use cases
- **Long-term**:  
  - Migrate all use cases
  - Deprecate desktop-based solution
When? - Factors for timing

- Benefits of migration
- Tool/use-case complexity
- Feature support on target platform
- Competitive pressure
- Technical pressure
- Ability to invest
- Development team know-how
How? - 3 key learnings

1. Migrate **iteratively**
2. Migrate **iteratively**
3. Migrate **iteratively**
Why migrate iteratively?

- Avoid “rewrite everything”
- Avoid high maintenance cost on legacy product
- Preserve investments made
- Onboard your development team
- Onboard your customers
- Discover their new requirements
Iterate use-case by use-case

- Start with “low hanging fruits”
  - High value
  - Low (additional) cost
- Add use-case by use-case opportunistically
- Create shippable product increments (MVP)
- Deploy product increments
Check product feasibility up-front

Build **Proof-Of-Concept** prototype for all key features

→ Shows feasibility
→ Learning experience for development team
→ Show weaknesses of target platform
Methods to support iterative migration

- Backporting
- Single-sourcing
- Declarative artifacts
- Extract services
- Build standalone
- Reuse
Backporting

- **When:** Short-term
- **Method:**
  - Develop feature web-based
  - Retrofit into desktop-based solution
- **Advantages:**
  - No “sunk cost”
  - Shared codebase
- **Risks/Challenges**
  - Interaction with other components (selection, DND)
  - Look-and-feel
Single-Sourcing

- **When:** Short-term
- **Method:** Share code-base
- **Advantages:**
  - Shared maintenance cost
  - No migration cost
- **Risks/Challenges**
  - Increased initial development cost
Declarative Artifacts

- When: Short- to medium-term
- Method: Extract information from code
- Advantages:
  - Declarative artifact can be shared
  - Lowers migration and maintenance cost
- Risks/Challenges
  - Expressiveness
  - Refactoring cost
Extract services

- **When:** Short- to medium-term
- **Method:** Extract business logic into (deployable) services
- **Advantages:**
  - Service can be shared
  - Lowers migration and maintenance cost
- **Risks/Challenges**
  - (Remote) performance
  - Refactoring cost
Standalone

- When: Medium- to long-term
- Method: Build standalone components
- Advantages:
  - Independent from web framework
  - Lowers technology risk
- Risks/Challenges
  - Higher initial development cost
Reuse

- **When:** Medium- to long-term
- **Method:** Reuse existing web-based components
- **Advantages:**
  - Lowers maintenance cost
  - Lowers technology risk
- **Risks/Challenges**
  - Availability
  - Integration
  - Usability
Key takeaways

- Thoroughly evaluate the benefits
- Migrate iteratively - use-case by use-case
- Consider proposed methods to ease iterative migration:
  - Backporting
  - Single-sourcing
  - Declarative artifacts
  - Extract services
  - Build standalone
  - Reuse

- Due Now:
  - Analyse use-cases to define if and when
  - Define a strategy
  - Build POC
Technical content on building web-based tools at ECE

- **Today**
  - Lucky in the Cloud With Diagrams - 03:15 PM - Bürgersaal 2
  - **Towards a Graphical LSP for Diagrams** 4:15 PM - Seminarraum 5
  - BoF: EMF Forms, JSON Forms,... - 07:30 PM - Seminarraum 5
- **Wednesday**
  - Building a Web-IDE based on Eclipse Theia... - 11:55 AM - Bürgersaal 2
  - EMF, JSON and I - 02:45 PM - Theater Stage
  - Eclipse Che: Theia IDE on steroids - 3:45 PM - Silchersaal
  - Domain-Specific Languages in the Cloud - 04:30 PM - Silchersaal
- **Thursday**
  - JSON Forms 2.0 - 10:45 AM - Theater Stage
  - Building Web-based Modeling Tools based ... - 11:30 AM - Theater Stage
  - The Busy RCP Developer’s Guide to Eclipse Theia - 01:15 PM - Theater Stage
Evaluate this Session

Sign in and vote at eclipsecon.org

WITH

-1 0 +1
## Technical details on building web-based tools

- What every Eclipse developer should know about Eclipse 4 (e4)  
  - Tuesday - 09:00 AM - Theater Stage

- Extending Eclipse Che for fun and profit  
  - Tuesday - 09:00 AM - Seminarräume 3-4

- If, when and how? - Strategies towards web-based tooling  
  - Tuesday - 02:30 PM - Bürgersaal 2

- Lucky in the Cloud With Diagrams  
  - Tuesday - 03:15 PM - Bürgersaal 2

- Towards a Graphical Language Server Protocol for Diagrams?  
  - Tuesday - 04:15 PM - Seminarraum 5

- Eclipse Kahoot!  
  - Tuesday - 06:30 PM - Bürgersaal 1

- BoF: EMF Forms, JSON Forms, EMF Client Platform, EDAPT and EMFStore  
  - Tuesday - 07:30 PM - Seminarraum 5

- Building a Web-IDE based on Eclipse Theia for Smart Home  
  - Wednesday - 11:55 AM - Bürgersaal 2

- Building a tool for engineers based on EMF  
  - Wednesday - 02:00 PM - Theater Stage

- Your custom "UML like" tool based on Papyrus  
  - Wednesday - 02:45 PM - Silchersaal

- EMF, JSON and I  
  - Wednesday - 02:45 PM - Theater Stage

- Eclipse Che: Theia IDE on steroids  
  - Wednesday - 03:45 PM - Silchersaal

- Modeling Symposium  
  - Wednesday - 04:30 PM - Bürgersaal 2

- Domain-Specific Languages in the Cloud – With Eclipse Technologies  
  - Wednesday - 04:30 PM - Silchersaal

- JSON Forms 2.0  
  - Thursday - 10:45 AM - Theater Stage

- Building Web-based Modeling Tools based on Eclipse Theia  
  - Thursday - 11:30 AM - Theater Stage

- The Busy RCP Developer’s Guide to Eclipse Theia  
  - Thursday - 01:15 PM - Theater Stage