Tool Support for Architectural Pattern Selection and Application in Cloud-Centric Service-Oriented IDEs

Fulya Horozal, Philip Reimer, Sebastian Scholze
ATB – Institut für angewandte Systemtechnik Bremen, Germany
horozal@atb-bremen.de
Software Architecture Design

- High-level structure of system components & their interactions
- High impact on quality, success & management of software
- Architectural patterns & styles
  - Principles & best practices for software architecture design
  - Guidelines & templates for structuring & organizing software systems
  - Common vocabulary to describe software architecture
  - E.g., event-driven architecture, layered architecture, microservices
Software Architecture Design

• Choosing the right architectural pattern
  • Strengths, drawbacks, technical knowledge
  • Impact on quality attributes (“-abilities”), requirements, constraints
  • Project requirements, constraints & limitations
  • System complexity, scalability needs
  • Team expertise, trade-offs
  • Industry best practices

• Traditional methods
  • Architectural pattern catalogs, architectural decision records
  • Architecture tradeoff analysis, decision matrices
  • Expert consultation, reference architectures
Architectural Decision Tool Support

- **Modeling and diagramming**
  - UML-based software modeling
  - Architectural diagrams
  - E.g., Enterprise Architect (Sparx Systems), IBM Rational Software Architect, Archimate Toolset, Eclipse Papyrus

- **Architectural decision modeling framework (O. Zimmermann)**

- **Quality attribute analysis**
  - Performance & security analysis
  - Scalability & maintainability assessment
  - Cost & change impact analysis
  - E.g., ARIS (Software AG), IBM Rational Rhapsody, QualiWare, Determine

- **ML techniques to learn from architectural decisions (Mahabaleshwar)**

- **Decision studio web tool for technology selection & architectural patterns (Farshidi et al.)**

- **Code generation from architectural models**
  - From UML or other notations
  - Scaffolding & project organization tools
A Framework for Architectural Pattern Selection and Application

- Architectural pattern decision support feature for IDE integration
- Architectural pattern selection
  - Knowledge base
    - Application domain
    - Application type
    - Quality attributes / non-functional requirements
    - Development & deployment requirements
    - Architectural features
  - Evaluation & ranking
    - Based on existing literature on pattern analyses (Farshidi et al. 2020, Richards 2022)
    - Scoring system assigning weights to patterns in context of knowledge base
- Architectural pattern application
  - GitHub repository templates for project & code organization
- Integrated into the cloud-native SmartCLIDE IDE
Supported Architectural Patterns

- Layered architecture
  - Distinct layers for presentation, application logic, data storage
- Event-driven architecture (EDA)
  - Systems communicate through events (trigger actions or reactions)
- Microkernel architecture
  - Essential core (the microkernel) and various optional modules
- Microservices architecture
  - Small independent services that communicate over APIs
- Service-oriented architecture (SOA)
  - Loosely coupled, reusable services communicating via interfaces
- Space-based architecture (SBA)
  - Distributes data & processing across a network of interconnected, distributed spaces
<table>
<thead>
<tr>
<th>Application Domain</th>
<th>Associated Architectural Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web-based systems</td>
<td>EDA, layered, microservices, SOA, SBA</td>
</tr>
<tr>
<td>Web services</td>
<td>Microservices, SOA, SBA</td>
</tr>
<tr>
<td>Service-based systems</td>
<td>Microservices, SOA</td>
</tr>
<tr>
<td>Distributed systems</td>
<td>EDA, layered, microkernel, microservices, SOA, SBA</td>
</tr>
<tr>
<td>Cloud computing applications</td>
<td>Microservices, SOA</td>
</tr>
<tr>
<td>Mobile applications</td>
<td>Layered, microservices, SOA, SBA</td>
</tr>
<tr>
<td>Compiler design</td>
<td>Layered</td>
</tr>
<tr>
<td>CASE and related developer tools</td>
<td>EDA, layered, microkernel, microservices,</td>
</tr>
<tr>
<td>Database systems</td>
<td>EDA, layered, microservices</td>
</tr>
<tr>
<td>Context-aware systems</td>
<td>EDA, layered, microservices, SOA</td>
</tr>
<tr>
<td>Adaptable systems</td>
<td>Microkernel, microservices</td>
</tr>
<tr>
<td>Enterprise application integration</td>
<td>EDA, microservices, SOA</td>
</tr>
<tr>
<td>Customer relationship management</td>
<td>EDA, layered, microservices, SOA</td>
</tr>
<tr>
<td>Information management and decision support system</td>
<td>EDA, layered, SOA</td>
</tr>
<tr>
<td>Transaction processing</td>
<td>EDA, layered, microservices, SOA</td>
</tr>
</tbody>
</table>
## Application Type

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Associated Architectural Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web application / website with small components</td>
<td>Microservices, SOA</td>
</tr>
<tr>
<td>Large scale web application like e-commerce or social website development</td>
<td>EDA, layered, microservices, SOA, SBA</td>
</tr>
<tr>
<td>General desktop application</td>
<td>Layered</td>
</tr>
<tr>
<td>Application with a simple business logic that does not need to scale out</td>
<td>EDA, layered</td>
</tr>
<tr>
<td>Enterprise or business application with traditional IT departments and processes</td>
<td>Layered, SOA</td>
</tr>
<tr>
<td>Application with fixed set of core functionalities and a dynamic set of functionalities that need frequent updates</td>
<td>Microkernel, microservices</td>
</tr>
<tr>
<td>Large, complex, enterprise-wide systems that require integration with many heterogeneous applications</td>
<td>EDA, microservices, SOA</td>
</tr>
<tr>
<td>Application with many shared components, particularly components across the enterprise</td>
<td>EDA, microservices, SOA</td>
</tr>
<tr>
<td>Application with immense and rapidly growing data systems</td>
<td>EDA, microservices, SBA</td>
</tr>
<tr>
<td>Application with different platforms</td>
<td>Microservices, SOA</td>
</tr>
<tr>
<td>Application that requires strict standards of testability</td>
<td>Layered</td>
</tr>
</tbody>
</table>
# Quality Attributes / NFRs

<table>
<thead>
<tr>
<th>Quality Attributes / Non-functional Requirements</th>
<th>Associated Architectural Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintainability</td>
<td>All six</td>
</tr>
<tr>
<td>Performance / Efficiency</td>
<td>EDA, microservices, SOA, SBA</td>
</tr>
<tr>
<td>Portability</td>
<td>All six</td>
</tr>
<tr>
<td>Reliability</td>
<td>All six</td>
</tr>
<tr>
<td>Security</td>
<td>All six</td>
</tr>
<tr>
<td>Development &amp; Deployment Requirements</td>
<td>Associated Architectural Patterns</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>High ease of development / quick development with fewer developers</td>
<td>Layered, microservices</td>
</tr>
<tr>
<td>Ease of rewriting and updating parts of the application</td>
<td>EDA, microkernel, microservices, SOA</td>
</tr>
<tr>
<td>Development teams that are spread out</td>
<td>Microservices</td>
</tr>
<tr>
<td>Adding special functionality, modules or extensions without modifying the original application</td>
<td>Microkernel, microservices</td>
</tr>
<tr>
<td>High ease of deployment</td>
<td>Microkernel, microservices</td>
</tr>
<tr>
<td>Rapid, frequent and independent deployment</td>
<td>Microservices</td>
</tr>
<tr>
<td>Quick response to a constantly changing environment</td>
<td>EDA, microkernel, microservices, SBA</td>
</tr>
<tr>
<td>Reusability of integrations and components sharing</td>
<td>EDA, microservices, SOA</td>
</tr>
<tr>
<td>Architectural Features</td>
<td>Associated Architectural Patterns</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Asynchronous communication / data flow</td>
<td>EDA, layered, microservices, SBA</td>
</tr>
<tr>
<td>Synchronous communication / data flow</td>
<td>Layered, microkernel, microservices, SOA</td>
</tr>
<tr>
<td>Loose coupling</td>
<td>EDA, microservices, SOA</td>
</tr>
<tr>
<td>Independent services</td>
<td>Microservices</td>
</tr>
<tr>
<td>Separation of concerns</td>
<td>Layered, microkernel, microservices, SOA</td>
</tr>
<tr>
<td>Plug-in components</td>
<td>Microkernel</td>
</tr>
<tr>
<td>Dynamic composition</td>
<td>EDA, microkernel, SOA, SBA</td>
</tr>
<tr>
<td>High volume data</td>
<td>EDA, microservices, SBA</td>
</tr>
</tbody>
</table>
Architectural Pattern Application

- 18 GitHub repository templates
- Frameworks: Java Spring Node.js Python
- Template for each architectural pattern & framework
Implementation

- Backend REST API in Java Spring
  - Retrieve survey content
  - Select architectural pattern
  - Select repository template
- **Independent of survey content & evaluation values**
  - JSON format for survey content & evaluation values
  - Reconfigurable
SmartCLIDE Project

- **H2020 EU-funded project (2020-2023)**
  - [https://smartclide.eu/](https://smartclide.eu/)
- **Novel cloud-native IDE**
  - [https://ide.che.smartclide.eu/](https://ide.che.smartclide.eu/)
  - Based on Eclipse Theia
  - Life cycle support (development, testing, deployment, run-time)
  - Collaborative discovery, creation, composition, testing, deployment of services in the cloud
  - Source code monitoring
  - CI/CD integration
- **4 industry pilots for validation & assessment**
  - Real-time communication platform (Wellness Telecom, Spain)
  - Social security application (Netcompany-Intrasoft, Luxembourg)
  - IoT web catalog (Unparallel, Portugal)
  - Project management solution (CONTACT Software, Germany)
- **Open sourced under Eclipse Foundation**
  - Eclipse OpenSmartCLIDE
### Workflows

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model import</td>
<td>1.0</td>
<td>22-Mar-2023 16:27</td>
</tr>
<tr>
<td>Model import</td>
<td>1.0</td>
<td>22-Mar-2023 16:20</td>
</tr>
<tr>
<td>Github API</td>
<td>1.0</td>
<td>21-Mar-2023 17:05</td>
</tr>
</tbody>
</table>

### Services

<table>
<thead>
<tr>
<th>Name</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>test-04</td>
<td>01-Sep-2023 16:18</td>
</tr>
<tr>
<td>test-03</td>
<td>31-Aug-2023 19:13</td>
</tr>
<tr>
<td>test-python-01</td>
<td>31-Aug-2023 13:26</td>
</tr>
</tbody>
</table>
Architectural Pattern Selection in SmartCLIDE IDE
Architectural Pattern Selection in SmartCLIDE IDE
Architectural Pattern Selection in SmartCLIDE IDE
Architectural Pattern Selection in SmartCLIDE IDE

Architectural Pattern Assistant

Please choose the desired features of your application for development and deployment:
- High ease of development / quick development with fewer developers
- Easy rewriting and updating parts of the application
- Development teams that are spread out
- Adding special functionality, modules or extensions without modifying the original application
- High ease of deployment
- Rigid, frequent and independent deployment
- Quick response to a constantly changing environment
- Reusability of integrations and component sharing

Please choose the desired features of your architecture:
- Asynchronous communication / data flow (interaction between components without strict requirement for immediate or synchronized responses)
- Synchronous communication / data flow (information can only be exchanged in real time)
- Loose coupling (degree of dependency between components is very low)
- Independent services (services can be developed and deployed independently of one another)
- Separation of concerns (separating an application into distinct sections each of which address a separate concern)
- Plug-in components (adding additional feature as plugins to the core application)
- Dynamic composition (system components and connections can be created and destroyed during runtime)
- High volume data (size of datasets to be processed are larger than terabytes)

According to your input, the most suitable patterns and corresponding scores are:
1. Microkernel (30)
2. Microservices (19)
3. Layered (14)
4. Event-driven (13)
5. Service oriented (13)
6. Space-based (12)
Architectural Pattern Selection in SmartCLIDE IDE

[Step 2/2] Service Details
Provide the details of the new service

Name
demo-project-01

Description
demo

Architectural Pattern
Microkernel

Framework
Python

Visibility
Private

License
Eclipse Public License 2.0

Cancel  Add
Future Work

- Increase # of patterns supported
- Support pattern combinations
- Improve survey content & evaluation
- Add explanation to pattern suggestions
- Add alternative structures to repository templates
eSAAM 2023
on Cloud to Edge Continuum

Sponsored by:

https://ide.che.smartclide.eu/

Organized by: