**Welke architectuur soort is het best toe te passen op het project?**

**Methods: Competitive Analysis/Available product analysis – Literature study**

Verschillende beschikbare architecturen bekeken en tegenover elkaar gezet

Uiteindelijk op basis van een matrix gekeken welk architectuur het best bij het project past.

**Een aantal Architectuur soorten en hun pros en cons/functionaliteiten:**

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Layered Architecture | Applicaties die snel gebouwd moeten zijn | Ongeorganiseerde broncode en modules met geen definitieve rollen kunnen een probleem vormen |
|  | Enterprise applicaties die traditionele IT-divisies en processen nodig hebben | Het overslaan van eerdere lagen om een strakke koppeling te creëren, kan leiden tot een logica puinhoop vol complexe onderlinge afhankelijkheden |
| Handig voor teams met weinig ervaring die niet veel weten over architectuur patronen | Basisaanpassingen kunnen een volledige redeployment van de applicatie vereisen |
| Applicaties die een strenge standaard aan onderhoud en testability nodig hebben |  |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Event-driven Architecture Pattern | Voor applicaties waar afzonderlijke gegevensblokken samenwerken met maar enkele modules. | Het testen van individuele modules kan alleen worden gedaan als ze onafhankelijk zijn, anders moeten ze worden getest in een volledig functioneel systeem |
|  | Helpt met user interfaces | Wanneer meerdere modules dezelfde events verwerken, wordt error handling een uitdaging om te structureren. |
|  | | De ontwikkeling van een systeembrede gegevensstructuur voor gebeurtenissen kan moeilijk worden als de events verschillende behoeften hebben. |
| Aan te raden voor een high performant Agile project |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Microkernel Architecture Pattern | Toepassingen met een duidelijke segmentatie tussen basisroutines en regels van hogere orde | De plug-ins moeten goede handshaking-code hebben, zodat de microkernel op de hoogte is van de installatie van de plug-in en klaar is om te werken |
|  | Toepassingen met een vaste set kernroutines en een dynamische set regels die regelmatig moeten worden bijgewerkt | Het veranderen van een microkernel is bijna onmogelijk als er meerdere plugins van afhankelijk zijn. |
|  | | Het is moeilijk om vooraf de juiste granulariteit (Hoe deelbaar een systeem is) voor de kernelfunctie te kiezen en later is het complexer. |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Microservices Architecture Pattern | Bedrijven en webapplicaties die een snelle ontwikkeling vereisen. | Het ontwerpen van het juiste granulariteitsniveau voor een servicecomponent is altijd een uitdaging. |
|  | Websites met kleine componenten, datacenters met goed gedefinieerde grenzen en teams op afstand wereldwijd. | Alle toepassingen bevatten geen taken die kunnen worden opgesplitst in onafhankelijke eenheden. |
|  | | De prestaties kunnen worden beïnvloed doordat taken worden verspreid over verschillende microservices. |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Space-Based Architecture | Applicaties en softwaresystemen die functioneren met een groot gebruikersbestand en een constante lading aanvragen. | Het is een complexe taak om de gegevens snel in de cache op te slaan zonder meerdere kopieën te storen. |
|  | Toepassingen die verondersteld worden schaalbaarheids- en gelijktijdigheidsproblemen aan te pakken. |  |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Client-Server Architecture | Applicaties zoals e-mails, online bankdiensten, het World Wide Web, afdrukken via netwerken, toepassingen voor het delen van bestanden, gaming-apps, etc. | Incompatibele servercapaciteit kan vertragen, waardoor een prestatieknelpunt ontstaat. |
|  | Applicaties die zich richten op real-time services zoals telecommunicatie-apps zijn gebouwd met een gedistribueerde applicatiestructuur. | Servers zijn meestal gevoelig voor een single point of failure. |
|  | Applicaties die gecontroleerde toegang vereisen en meerdere services bieden voor een groot aantal gedistribueerde clients. | Het veranderen van het patroon is een complex en duur proces. |
|  | Een applicatie met gecentraliseerde bronnen en services die over meerdere servers moet worden gedistribueerd. | Serveronderhoud kan een veeleisende en dure taak zijn. |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Master-Slave Architecture | Ontwikkeling van besturingssystemen die mogelijk een architectuur met meerdere processors vereisen. | Falen van de mastercomponent kan leiden tot een verlies van gegevens zonder back-up over de slave-componenten. |
|  | Geavanceerde toepassingen waarbij grotere diensten moeten worden opgesplitst in kleinere componenten. | Afhankelijkheden binnen het systeem kunnen leiden tot een storing van de slave-componenten. |
|  | Toepassingen die onbewerkte gegevens verwerken die zijn opgeslagen op verschillende servers via een gedistribueerd netwerk. | Er kan een toename van overheadkosten zijn als gevolg van het geïsoleerde karakter van de slave-componenten. |
|  | Webbrowsers die multithreading volgen om de responsiviteit te vergroten. |  |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Pipe-Filter Architecture | Het kan worden gebruikt voor toepassingen die een eenvoudige, eenrichtingsverkeersverwerking en -transformatie mogelijk maken. | Er kan een verlies van gegevens tussen filters zijn als het infrastructuurontwerp niet betrouwbaar is. |
|  | Toepassingen met behulp van tools zoals Electronic Data Interchange en External Dynamic List. | Het langzaamste filter beperkt de prestaties en efficiëntie van de gehele architectuur. |
|  | Ontwikkeling van data compilers die worden gebruikt voor foutcontrole en syntax analyse. | Tijdens de overdracht tussen filters kunnen de overheadkosten voor gegevenstransformatie toenemen. |
|  | Om geavanceerde bewerkingen uit te voeren in besturingssystemen zoals UNIX, waarbij de uitvoer en invoer van programma's in een reeks zijn verbonden. | Het continue transformationele karakter van de architectuur maakt het minder gebruiksvriendelijk voor interactionele systemen. |
|  |  | |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Broker Architecture Pattern | Gebruikt in message broker software zoals Apache ActiveMQ, Apache Kafka, RabbitMQ en JBoss Messaging. | Lage fouttolerantiecapaciteit |
|  | Voor het structureren van gedistribueerde systemen met ontkoppelde componenten. | Vereist standaardisatie van de servicebeschrijving. |
|  | | De verborgen laag kan de softwareprestaties verminderen. |
| Hogere latentie en vereist meer inspanning bij de implementatie. |

|  |  |  |
| --- | --- | --- |
| Architectuur | Gebruik | Tekortkomingen |
| Peer-to-Peer Architecture | Netwerken voor het delen van bestanden zoals Gnutella en G2 | Geen garantie voor een hoogwaardige service. |
|  | Op cryptocurrency gebaseerde producten zoals Bitcoin en Blockchain. | Het bereiken van robuuste beveiliging is een uitdaging. |
|  | Multimediaproducten zoals P2PTV en PDTP. | De prestaties zijn afhankelijk van het aantal knooppunten dat op het netwerk is aangesloten. |
|  | | Geen manier om een back-up te maken van bestanden of mappen. |
| Mogelijk heeft u een specifieke interface nodig om het bestand te lezen. |

### Architecture Matrix

**De matrix geeft een score aan elke architectuur om zo een goede vergelijking te kunnen maken:**

High Medium Doubt Low

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Agile | Development | Testing | Performance | Scaling |
| Layered |  |  |  |  |  |
| Event-Driven |  |  |  |  |  |
| Microkernel |  |  |  |  |  |
| Microservices |  |  |  |  |  |
| Spaced |  |  |  |  |  |
| Client-Server |  |  |  |  |  |
| Master-Slave |  |  |  |  |  |
| Pipe-Filter |  |  |  |  |  |
| Peer-to-Peer |  |  |  |  |  |

Aangezien schaalbaarheid een grote factor (zelfs een leeruitkomst) is bij het kiezen van een architectuur stijl, is Microkernel als enige echt geen optie. Voor dit project kies in een Microservice architectuur. Ik heb de meeste en makkelijkste toegang tot de microservice documentatie omdat er al veel over te vinden is in canvas.