Explicación de la arquitectura propuesta:

1. **App Flutter**:
   * Interfaz de usuario móvil/web que se comunica con el backend a través de API RESTful.
2. **API Gateway (Quarkus)**:
   * Punto de entrada único para todas las solicitudes de la app Flutter.
   * Maneja la autenticación y autorización (integrado con Keycloak).
   * Enruta las solicitudes a los microservicios apropiados.
   * Puede implementar políticas de throttling, caching, y transformación de datos.
3. **Microservicios Quarkus**:
   * Cada microservicio se encarga de un dominio específico de la lógica de negocio.
   * Implementados en Quarkus para aprovechar su rendimiento y baja huella de memoria.
   * Se comunican con los servicios SOAP existentes.
4. **Adaptadores SOAP**:
   * Cada microservicio Quarkus actúa como un adaptador para uno o más servicios SOAP.
   * Convierten las solicitudes REST en llamadas SOAP y viceversa.
5. **Servicios SOAP existentes**:
   * Servicios backend existentes que exponen funcionalidades a través de SOAP.
6. **Keycloak**:
   * Proporciona gestión de identidad y acceso centralizada.
   * Emite tokens JWT para autenticación y autorización.

Esta arquitectura es adecuada para tu proyecto por las siguientes razones:

1. **Desacoplamiento**: Los microservicios Quarkus actúan como una capa de abstracción entre tu moderna app Flutter y los servicios SOAP legacy. Esto permite evolucionar y modernizar tu sistema gradualmente.
2. **Escalabilidad**: Puedes escalar independientemente cada componente según sus necesidades específicas.
3. **Rendimiento**: Quarkus es conocido por su rápido tiempo de inicio y bajo consumo de memoria, lo que lo hace ideal para un middleware eficiente.
4. **Flexibilidad**: Puedes añadir, modificar o reemplazar microservicios sin afectar todo el sistema.
5. **Seguridad centralizada**: Keycloak proporciona una robusta capa de seguridad que puede ser aplicada consistentemente en toda la arquitectura.
6. **API Gateway**: Simplifica la interacción de la app Flutter con el backend, proporcionando un único punto de entrada y manejando preocupaciones transversales.
7. **Transformación de datos**: Los microservicios Quarkus pueden manejar la transformación entre JSON (para la app Flutter) y XML (para los servicios SOAP) de manera eficiente.

Para implementar esta arquitectura:

1. Utiliza Quarkus para crear el API Gateway y los microservicios.
2. Implementa clientes SOAP en los microservicios Quarkus usando extensiones como quarkus-cxf para consumir los servicios SOAP.
3. Utiliza RESTEasy en Quarkus para exponer endpoints RESTful que la app Flutter consumirá.
4. Integra Keycloak con Quarkus para la autenticación y autorización.
5. En la app Flutter, usa paquetes como dio o http para realizar llamadas HTTP al API Gateway.

Esta arquitectura te permite aprovechar las ventajas de Quarkus (rendimiento, facilidad de desarrollo) mientras mantienes la compatibilidad con tus servicios SOAP existentes, proporcionando una API moderna y eficiente para tu app Flutter.

¿Te gustaría que profundice en algún aspecto específico de esta arquitectura o en cómo implementar alguna parte en particular?