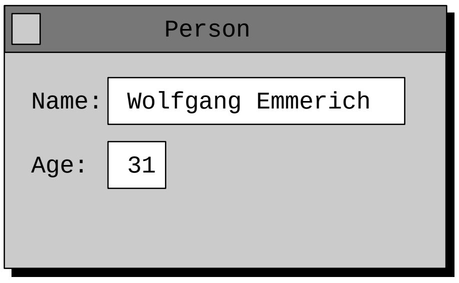
Genericity

Generic Applications

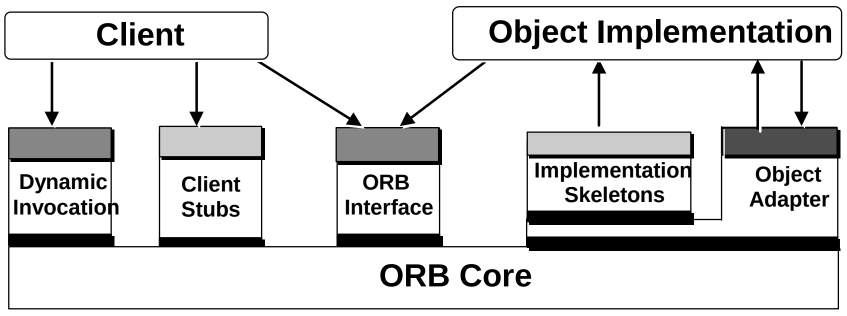
Generic applications use components whose types are not (yet) known.

Example: Object Browser



Static vs. Dynamic Invocation

- **_Example: OMG/CORBA_**



Static Invocation

- **_Advantages:_**
  - **_Requests are simple to define._**
  - **_Availability of operations checked by programming language compiler._**
  - **_Requests can be implemented fairly efficiently._**
- **_Disadvantages:_**
  - **_Generic applications cannot be build._**
  - **_Recompilation required after operation interface modification._**

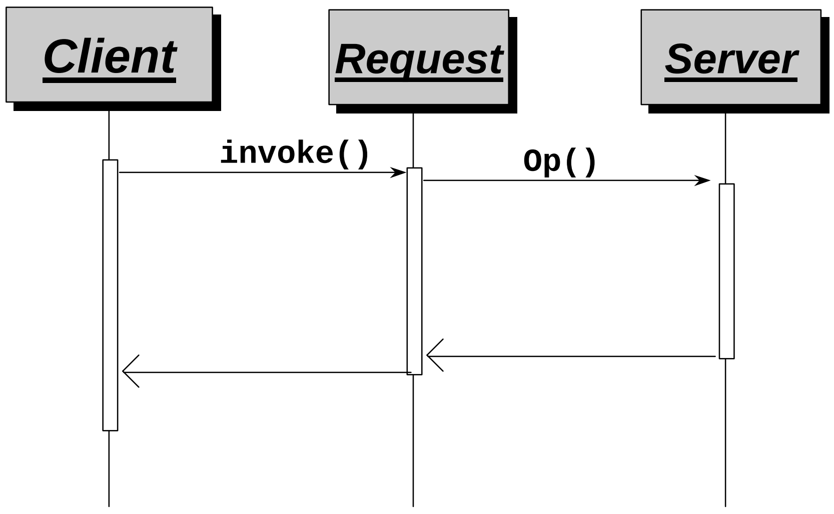
Dynamic Invocation

- ***Interface to create operation execution requests dynamically.***
- ***Requests are objects.***
- ***Attributes for operation name, parameters and results.***
- ***Operations to***
  - ***change operation parameters,***
  - ***issue the request and***
  - ***obtain the request results.***

Creation of Requests

```
interface Object {
 ORBstatus create_request (
  in Context ctx,         // operation context
  in Identifier operation,// operation to exec
  in NVList arg_list,     // args of operation
  inout NamedValue result,// operation result
  out Request request     // new request object
  in Flags req_flags      // request flags
  );
  ...
};
```

Synchronous Request



Dynamic Invocation

■ *Advantages:*
- ***Components can be built without having the interfaces they use,***
- ***Higher degree of concurrency through deferred synchronous execution.***
- ***Components can react to changes of interfaces.***

■ *Disadvantages:*
- ***Less efficient,***
- ***More complicated to use and***
- ***Not type safe!***

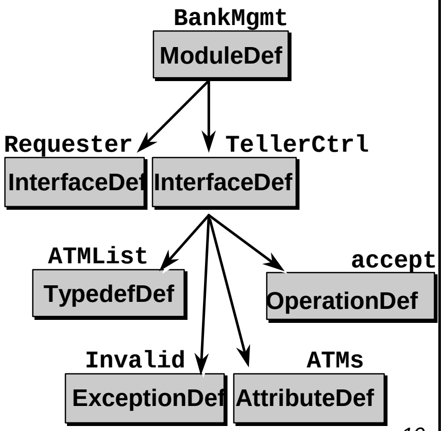
Interface Repository

- *Makes type information of interfaces available at run-time.*
- *Enables development of generic applications.*
- *Achieves type-safe dynamic invocations.*
- *Supports construction of interface browser.*
- *Used by Middleware itself.*

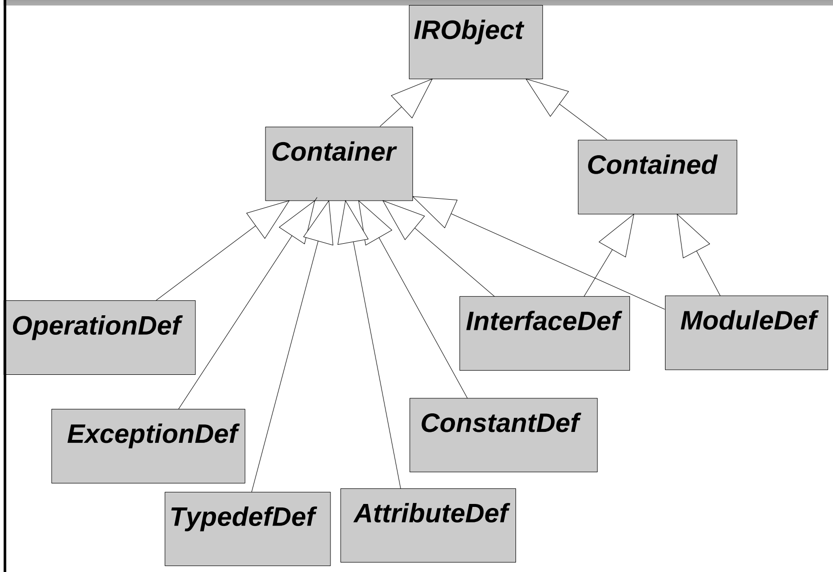
Abstract Syntax Trees (ASTs)

- *Interface repository persistently stores ASTs of IDL modules, interfaces, types, operations etc.*

```
module BankMgmt {
 interface Requester;
 interface TellerCtrl {
  typedef sequence<ATM>
          ATMList;
  exception Invalid {};
  attribute ATMList ATMs;
  void accept(
    in Requester req,
    in short amount);
 };
};
```



AST Node Types



Container (node with children)

```
interface Container : IRObject {
  Contained lookup(in ScopedName search_name);
  sequence<Contained> contents(
      in DefinitionKind limit_type,
      in boolean        exclude_inherited);

  sequence<Contained> lookup_name(
      in Identifier     search_name,
      in long           levels_to_search,
      in DefinitionKind limit_type,
      in boolean        exclude_inherited);
  ...
};
```

Contained (child)

```
interface Contained : IRObject {
  attribute Identifier         name;
  attribute RepositoryId       id;
  attribute VersionSpec        version;
  readonly attribute Container defined_in;
  struct Description {
    DefinitionKind  kind;
    any             value;
  };
  Description describe();
  ...
};
```

Interface Definition

```
interface InterfaceDef : Container,Contained {
 attribute sequence<InterfaceDef> base_interfaces;
 boolean is_a(in RepositoryId interface_id);
 struct FullInterfaceDescription {
    Identifier                       name;
    RepositoryId                     id;
    RepositoryId                     defined_in;
    RepositoryIdSequence             base_interfaces;
    sequence<OperationDescription> operations;
    sequence<AttributeDescription> attributes;
    ...
 };
 FullInterfaceDescription describe_interface();
};
```

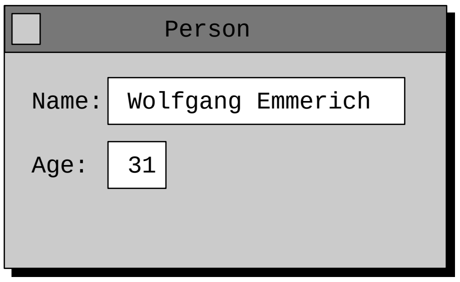
Locating Interface Definitions

Alternatives:

- ***Any interface inherits the operation InterfaceDef `get_interface()` from `Object`.***
- ***Associative search using `lookup_name`.***
- ***Navigation through the interface repository using `contents` and `defined_in` attributes.***

Example: Object Browser

- ***Use interface repository to find out about object types at run-time***
- ***Use dynamic invocation interface to obtain attribute values***

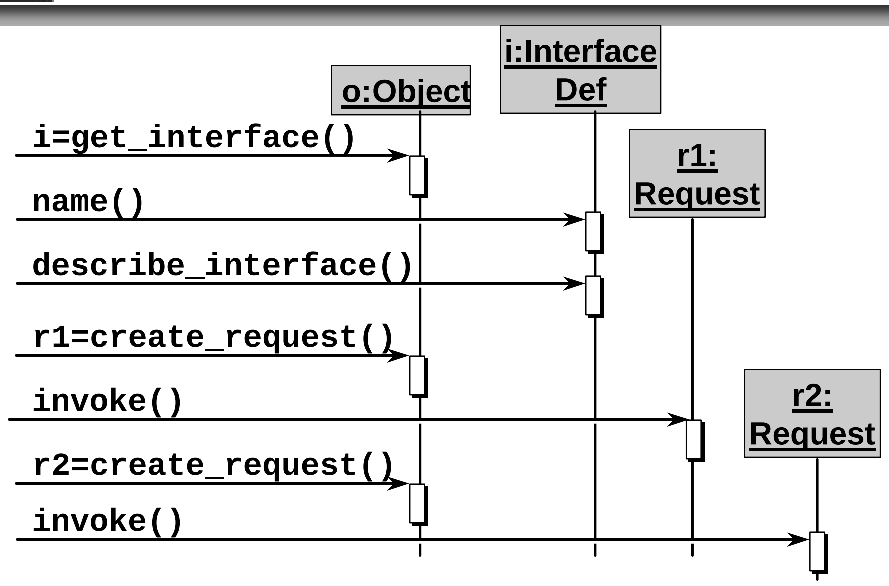


Person

Name: Wolfgang Emmerich

Age: 31

Sequence Diagram


o:Object
i:Interface Def
r1: Request
r2: Request
i=get_interface()
name()
describe_interface()
r1=create_request()
invoke()
r2=create_request()
invoke()