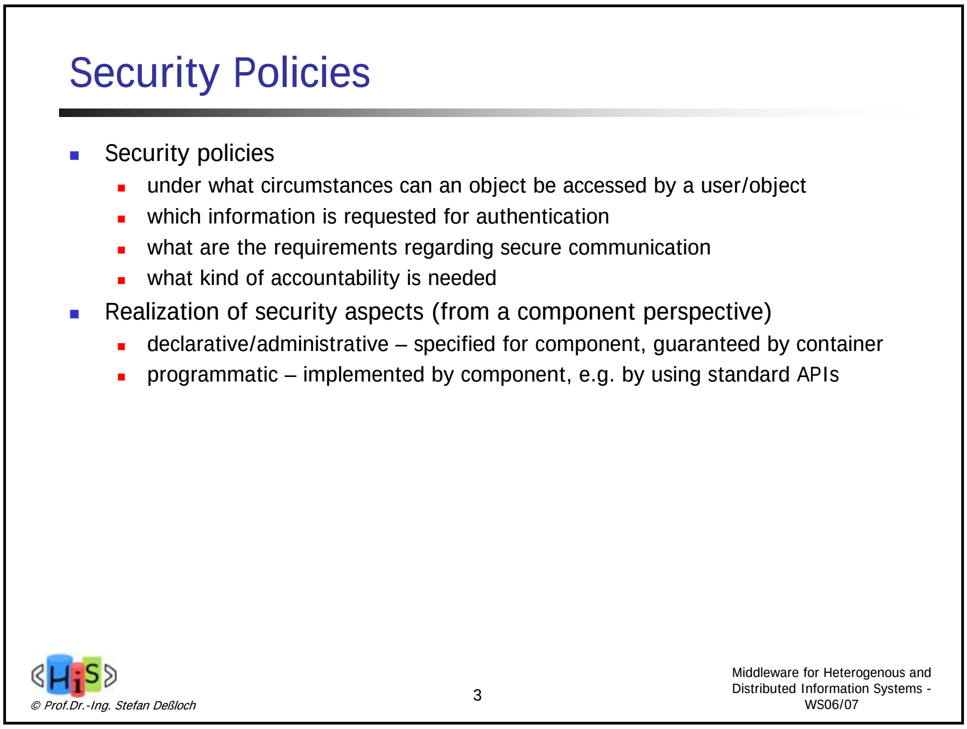
Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Chapter 7
Security and Connectors

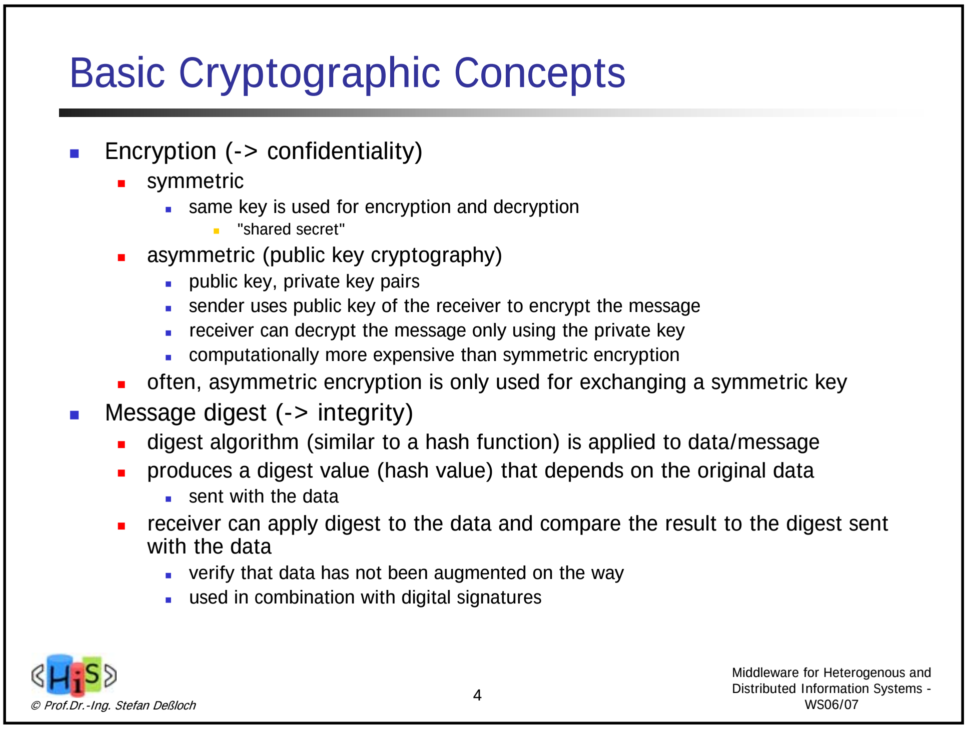
Key Security Features

- Identification/authentication: provide/verify proof of identity of *principals* (i.e., human user or application object)
  - user-id, password
  - certificate
- Authorization/access control: decide whether a principal can access a particular resource/object
- Communication security
  - Confidentiality: protection against eavesdroppers
  - Integrity: message was not modified accidentally or deliberately in transit
- Auditing: log information to make users accountable for their actions
  - record security-related events to detect and investigate security violations
- Non-repudiation: log information so that a principal cannot deny receiving or sending data/messages



Security Policies

- Security policies
  - under what circumstances can an object be accessed by a user/object
  - which information is requested for authentication
  - what are the requirements regarding secure communication
  - what kind of accountability is needed
- Realization of security aspects (from a component perspective)
  - declarative/administrative – specified for component, guaranteed by container
  - programmatic – implemented by component, e.g. by using standard APIs



Basic Cryptographic Concepts

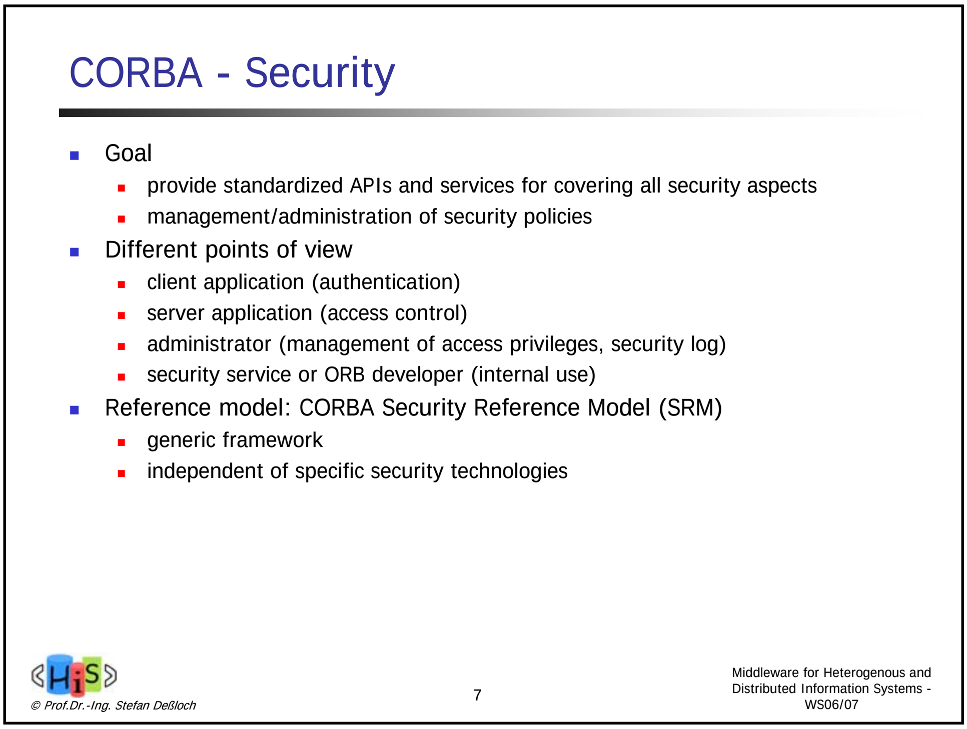
- Encryption (-> confidentiality)
  - symmetric
    - same key is used for encryption and decryption
      - "shared secret"
  - asymmetric (public key cryptography)
    - public key, private key pairs
    - sender uses public key of the receiver to encrypt the message
    - receiver can decrypt the message only using the private key
    - computationally more expensive than symmetric encryption
  - often, asymmetric encryption is only used for exchanging a symmetric key
- Message digest (-> integrity)
  - digest algorithm (similar to a hash function) is applied to data/message
  - produces a digest value (hash value) that depends on the original data
    - sent with the data
  - receiver can apply digest to the data and compare the result to the digest sent with the data
    - verify that data has not been augmented on the way
    - used in combination with digital signatures

Basic Cryptographic Concepts (cont.)

- Digital signature (-> integrity, authentication, non-repudiation)
  - The digest is encrypted with the private key of the signer, producing the signature
  - To verify the signature, anyone with access to the public key of the signer can
    - Decrypt the signature (original hash) using the public key
    - Apply the hash function to the original data
    - Compare the two hash values to make sure they are identical
  - Allows to make sure that
    - the data has not been modified
    - the data was actually sent by the owner of the public key
- Certificate
  - Data structure that holds at least the following information
    - identification (name, address, ...) of the certificate owner (person, company)
    - public key of the certificate owner
  - Issued by a certificate issuing authority
    - authority signs the certificate with its own private key

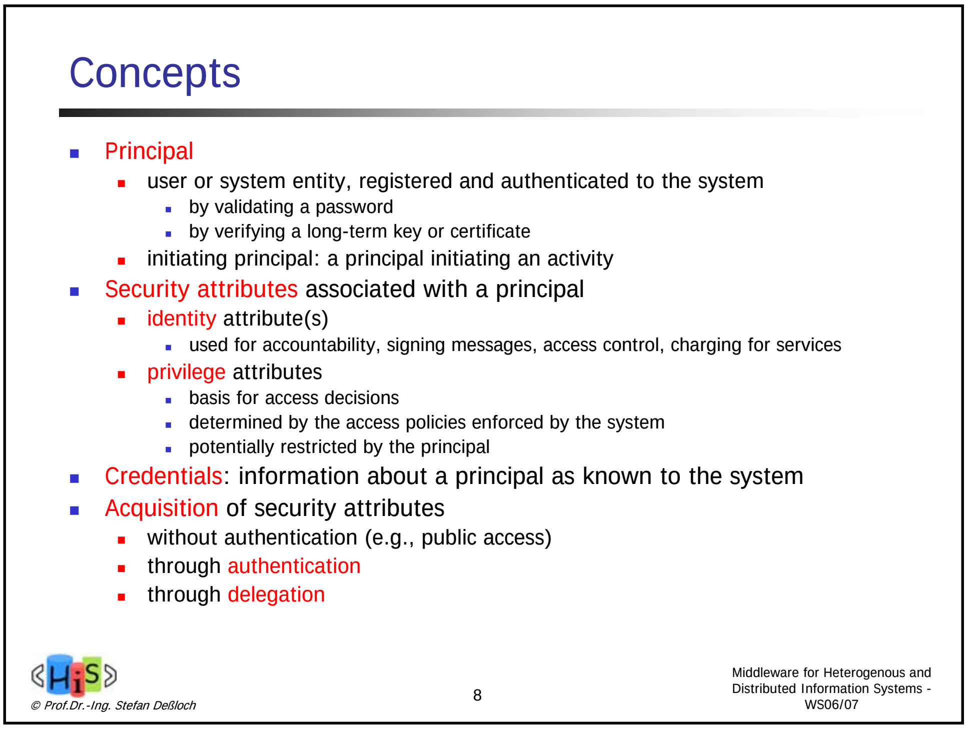
Transport Security

- HTTP Basic Authentication
  - UserID, Password authentication on the web
    - Initial HTTP request results in error “401 Unauthorized”
    - Browser opens dialog to request user, password info, resubmits the request
      - Userid/password are encoded in Base64, NOT encrypted
  - Web server verifies permissions based access control list (ACL)
- Secure Sockets Layer/Transport Layer Security (SSL/TLS)
  - Protocol for transmitting data in a secure way
    - point-to-point secure sessions
    - Can provide confidentiality, authentication, integrity
  - Located between application layer and transport layer (TCP)
    - Other protocols can be performed over SSL
      - HTTPS is HTTP over SSL
  - Supports server authentication and client authentication via certificates
    - The latter is rarely used, requires client to possess a certificate issued by a certificate authority
      - HTTP authentication frequently used here



CORBA - Security

- Goal
  - provide standardized APIs and services for covering all security aspects
  - management/administration of security policies
- Different points of view
  - client application (authentication)
  - server application (access control)
  - administrator (management of access privileges, security log)
  - security service or ORB developer (internal use)
- Reference model: CORBA Security Reference Model (SRM)
  - generic framework
  - independent of specific security technologies



Concepts

- Principal
  - user or system entity, registered and authenticated to the system
    - by validating a password
    - by verifying a long-term key or certificate
  - initiating principal: a principal initiating an activity
- Security attributes associated with a principal
  - identity attribute(s)
    - used for accountability, signing messages, access control, charging for services
  - privilege attributes
    - basis for access decisions
    - determined by the access policies enforced by the system
    - potentially restricted by the principal
- Credentials: information about a principal as known to the system
- Acquisition of security attributes
  - without authentication (e.g., public access)
  - through authentication
  - through delegation

Secure Object Invocation

- Establishing a security association
  - mutual authentication
  - making client credentials available to the server
    - *Credential* object
      - security attributes: identity, role (e.g., administrator), group, authorization level (e.g., confidential), *capabilities* (right to invoke specific methods on an object) ...
  - establishing a security context
  - ensure communication security
- Secure protocols
  - Secure Inter-ORB-Protocol (SECIOP)
  - SSL/TLS
- Access control, logging
  - client-side and server-side
  - performed by ORB, possibly by application

Authorization

- Object invocation access policies
  - system decides whether a client acting on behalf of a principal can invoke the requested operation
  - client- and server-side access decision functions apply access control rules based on
    - security id and privilege attributes of the initiator
      - role (e.g., administrator), group, authorization level (e.g., confidential), *capabilities* (right to invoke specific methods on an object) ...
    - control attributes of the target,
      - describe *policy details* for the respective target objects
        - *Access Control Lists* (ACLs)
        - *Labels* (confidential, ...)
    - addtl. context information
- Application access policies
  - application controls access, implements access decision functions

Delegation

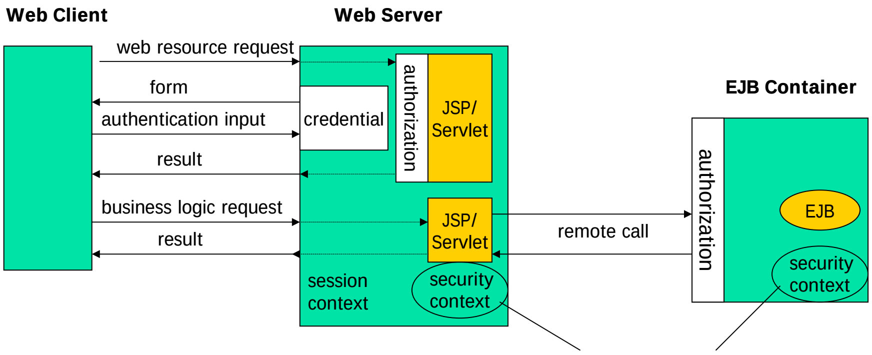
- Object request may result in an invocation/request chain
  - what privileges should we associated with 'intermediate' objects in the request chain?
- Delegation policy types
  - no delegation
    - intermediate uses its own privilege attributes
  - simple delegation or impersonation
    - intermediate uses invokers privilege attributes
  - combined privileges delegation
    - invokers privilege attributes are 'merged' into the intermediate privilege attributes
  - composite delegation
    - credentials of both the invoker and the intermediate are passed to the target
  - traced delegation
    - composite delegation which carries delegation chain history

CORBA Security – Conformance Levels

- Level 1: security is transparent for application
  - secure ORB
  - user (*principal*) is authenticated by the system
  - access control
  - secure communication
  - logging
- Level 2: application explicitly uses security service
  - client
    - authentication of users
    - controlled delegation of privileges
  - server
    - manipulation of privileges
    - authentication and authorization checking

J2EE Server Security

- Example scenario



establishing security association
based on credentials

Container-based Security

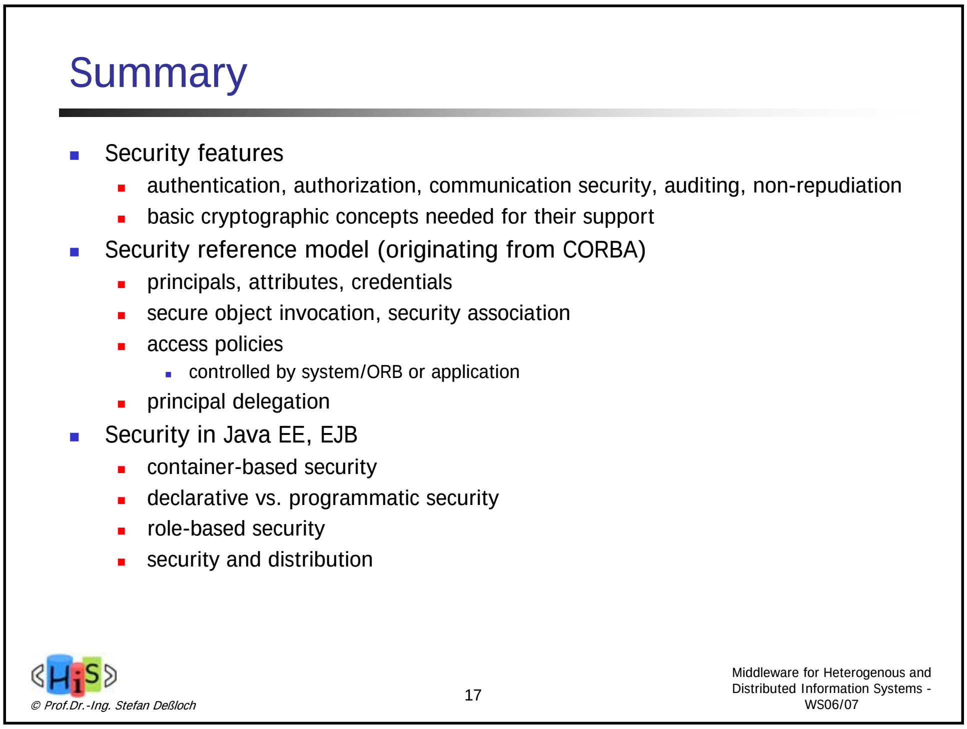
- Declarative security
  - deployment descriptor supports specification of security aspects
    - security roles
    - access control
    - authentication requirements
  - mapping of security aspects to the security environment of J2EE server during deployment phase
- Programmatic security
  - application components implement security aspects
    - may be required for realizing access control for individual instances based on conditions
      - example: role "clerk" can invoke Loan.approve() only if Loan.amount < 10000
  - standardized interfaces
    - web components
      - isUserInRole (HttpServletRequest)
      - getUserPrincipal (HttpServletRequest)
    - EJB components
      - isCallerInRole (EJBContext)
      - getCallerPrincipal (EJBContext)

Authorization in EJB

- Based on security roles
  - logical group of users
    - e.g. administrator, readOnly, ...
  - defined in deployment descriptor of each component (by application provider)
    - security-role
    - method-permission
      - defines roles allowed to invoke a method, if declarative security is used
  - mapped to user (group) during deployment phase
    - security-role -> user group(s), user identities
- Credentials used for access control
  - name – if role is mapped to individual users
  - group attribute – if role is mapped to user group
- Delegation
  - simple delegation is the default
  - arbitrary role name may be designated as the new principal
    - "runAs"-property

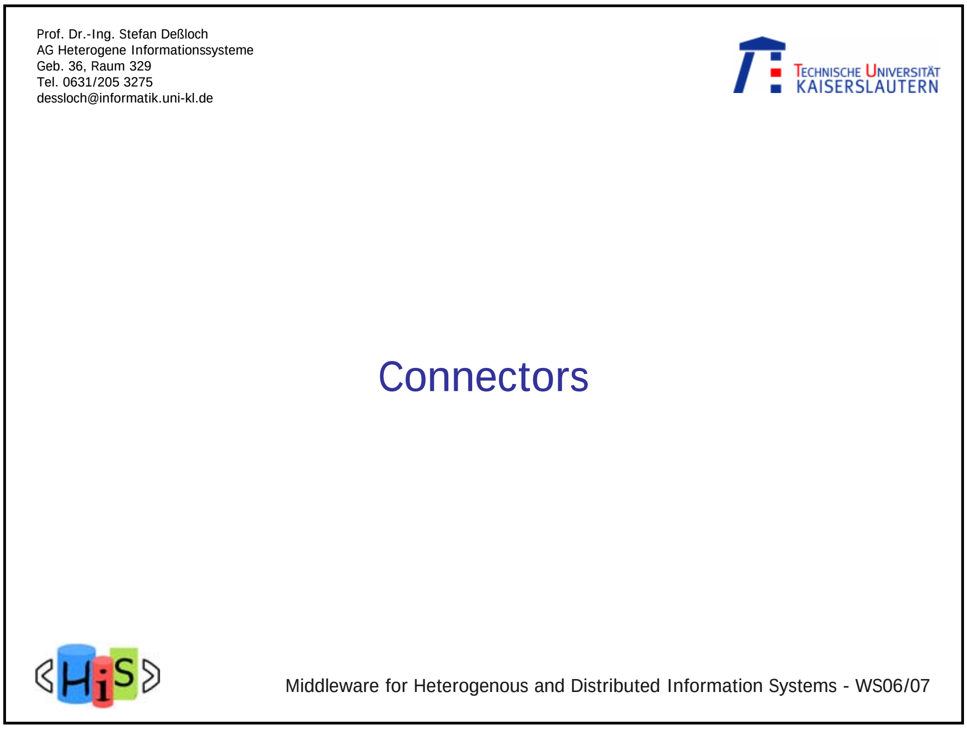
Security and Distribution

- Requires establishing security association
  - secure communication
  - joint security context
- Established automatically by participating containers
  - usually proprietary formats and protocols in the scope of single J2EE server product
  - standardized security information exchange in distributed, heterogeneous environments based on CORBA interoperability specifications
- Involving resource managers requires more complex measures
  - authentication across security policy domains
    - preconfigured security identity
    - programmatic authentication
    - additional capabilities are desirable (principal mapping, caller impersonation, credentials mapping)
  - additional capabilities defined in the scope of the J2EE Connector Architecture



Summary

- Security features
  - authentication, authorization, communication security, auditing, non-repudiation
  - basic cryptographic concepts needed for their support
- Security reference model (originating from CORBA)
  - principals, attributes, credentials
  - secure object invocation, security association
  - access policies
    - controlled by system/ORB or application
  - principal delegation
- Security in Java EE, EJB
  - container-based security
  - declarative vs. programmatic security
  - role-based security
  - security and distribution



Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

Technische Universität Kaiserslautern

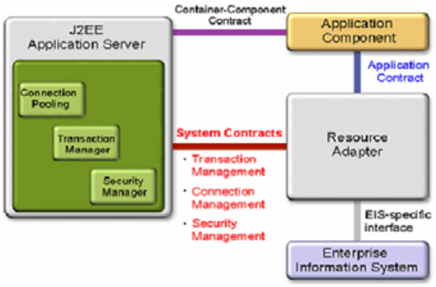
Connectors

Accessing Enterprise Information Systems

- Accessing (SQL) data bases is based on standardized (DB-gateway) interfaces
  - e.g., SQL + JDBC/SQLJ, or EJB CMP
  - interoperability at the system level supported through well-defined interfaces
    - XXXDataSource for Connection Pooling, transaction coordination, ...
- Accessing/interacting with enterprise information systems?
  - Examples
    - Enterprise Resource Planning (ERP), Customer Relationship Management (CRM)
      - SAP, Baan, Peoplesoft, Siebel, Oracle, ...
    - Transactional systems based on TP-monitors
      - CICS, Encina, Tuxedo, ...
    - Non-relational DBS
      - IMS, ...

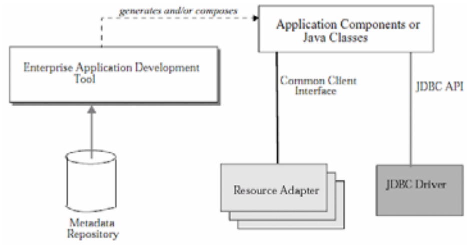
J2EE Connector Architecture (JCA)

- Standardized Interoperability with EIS
- Resource Adapter (Connector)
  - EIS-specific component
  - implements client interface (application contract) for EIS, used by EJBs, web components
    - either standardized (Common Client Interface, CCI)
    - or EIS-specific
  - cooperates with J2EE application server via system-level contracts
    - connection management, transactions, security, ...

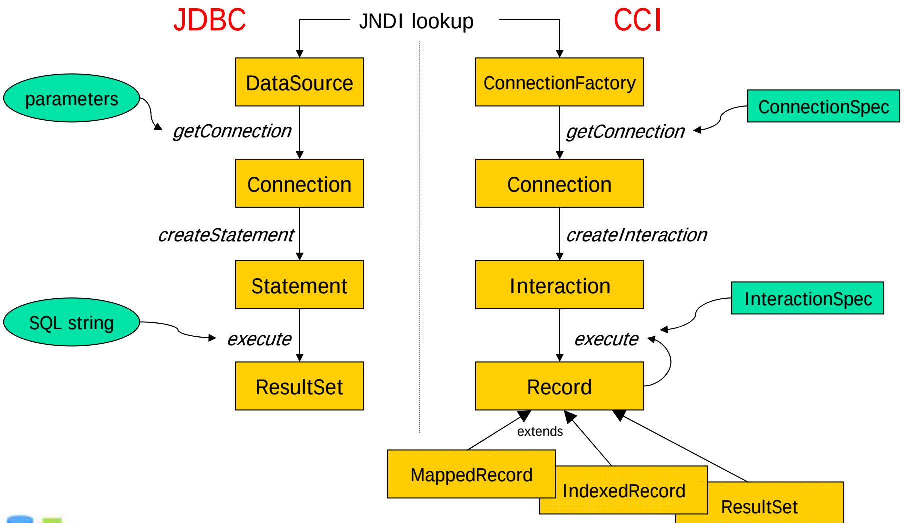


Common Client Interface

- Generic interface for invoking EIS functions (remote function calls)
- Useful for application development tooling, EAI frameworks
  - generating EJB wrapper classes for EIS functions (similar to EJB CMP tooling)
  - required standardized representation of EIS meta data, in addition to CCI
    - not defined as part of the JCA


generates and/or composes
Application Components or Java Classes
Enterprise Application Development Tool
Common Client Interface
JDBC API
Resource Adapter
JDBC Driver
Metadata Repository

CCI vs. JDBC Interfaces



JDBC

JNDI lookup

CCI

parameters

DataSource

getConnection

Connection

createStatement

Statement

SQL string

execute

ResultSet

ConnectionFactory

ConnectionSpec

getConnection

Connection

createInteraction

Interaction

InteractionSpec

execute

Record

extends

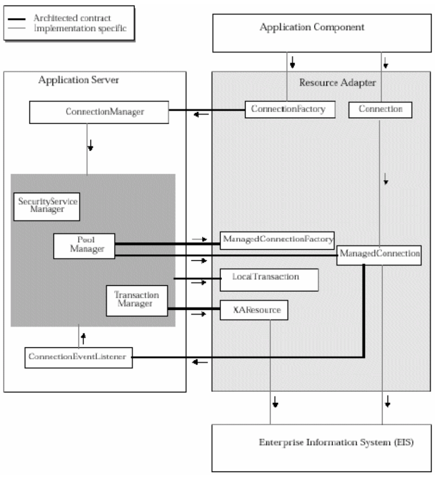
MappedRecord

IndexedRecord

ResultSet

JCA System-Level Contracts

- Application server implements ConnectionManager
  - generic, for arbitrary EIS connections
  - interacts with other AS services
    - connection pooling, transactions, security
- Resource Adapter (RA) creates connections (ConnectionFactory)
- Application connection request flows via ConnectionFactory to ConnectionManager
  - PoolManager selects suitable connection in the pool or initiates creation of a new connection
    - RA helps with selection process
  - RA informs ConnectionManager about connection state (ConnectionEventListener)



JCA – Transaction Management

- Resource Adapter (or RM of EIS) may support
  - global transactions
    - coordinated by TA manager of application server
    - XA-compliant (RA implements XAResource interface)
      - one-phase optimization possible, if only one resource is involved
  - local transactions
    - permits bypassing global TA manager for performance reasons, if it is known at deployment time that global TAs are not required
  - no transactions

JCA – Security

- Security architecture supports alternatives for determining the so-called *resource principal*
  - component-managed sign-on
    - application component determines resource principal (e.g. dynamically)
    - container-managed sign-on
    - resource principal, sign-on information (e.g. userid, password) defined for EIS at deployment time
      - configured identity: resource principal fixed, independent of initiating principal
      - principal mapping: resource principal determined based on initiating principal through a mapping, does not inherit any additional security attributes from initiating principal
      - caller impersonation: identity, credentials of caller are delegated to EIS
      - credentials mapping: mapping across security domains
- Choice of authentication mechanisms
  - BasicPassword, KerbV5, ...
- Access control can be performed by EIS or application server
- Secure communication supported by establishing security association with RA

Connectors - Summary

- Goal: Integration of existing EIS as additional resource managers in distributed component-based environments through a so-called resource adapter
  - unified connection/security model for calling application components
  - uniform interface to arbitrary EIS
    - tooling support, combined with meta data management
- Standardized interfaces, interactions
  - same RA can be installed in any J2EE-conforming server implementation
  - J2EE server realizes the required infrastructure only once, for arbitrary EIS
- Important architecture concept for Enterprise Application Integration
  - numerous J2EE connectors are commercially available