Outline

- Backgroud
- NR's Framework for Enforcement of Privacy Policies
- EPAL
- Java implementation of framework

Privacy research

- Computational disclosure control
- Communications privacy
- PETs
- **Enforcement of privacy policies**
- **Privacy and DRM**
- Ontologies/Vocabularies
- Policy languages
- Privacy and Identity Management (IM)
- Cryptographic primitives
- Economic models of privacy
- What is privacy?
- ….

OECD Principles

- Openness
- Purpose specification
- Collection limitation
- Use limitation
- Data quality
- Individual participation
- Accountability
- Security safeguards

What is personal data?

- Personal data is any information that identifies or can be used to identify, contact, or locate the person to whom such information pertains
- Additionally, any data linked to an identifiable individual is considered personal data

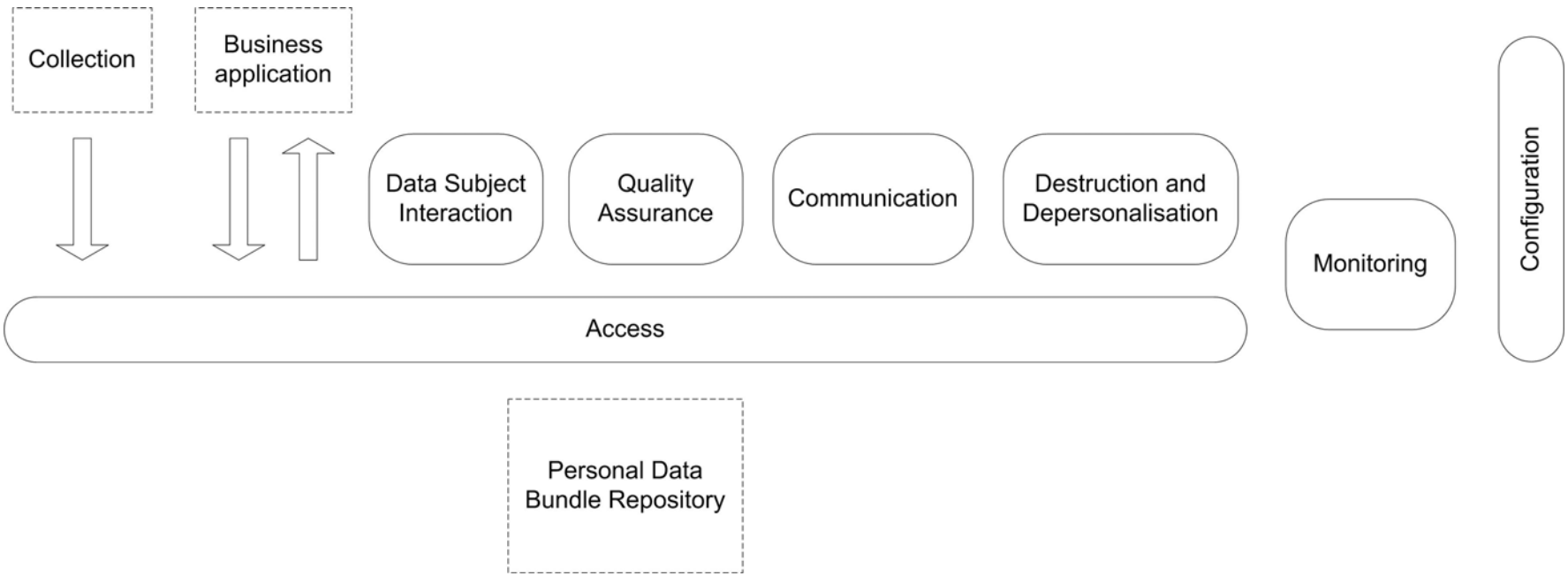
More definitions

- A *data subject* is a person that a set of personal data pertains to.
- A *data collector* is an organisation that collects, stores and processes personal data.
- A *data processor* is an organisation that processes personal data.

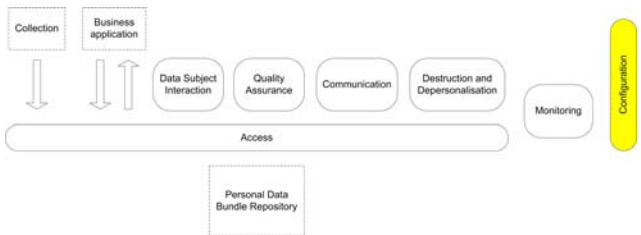
Approaches to privacy protection

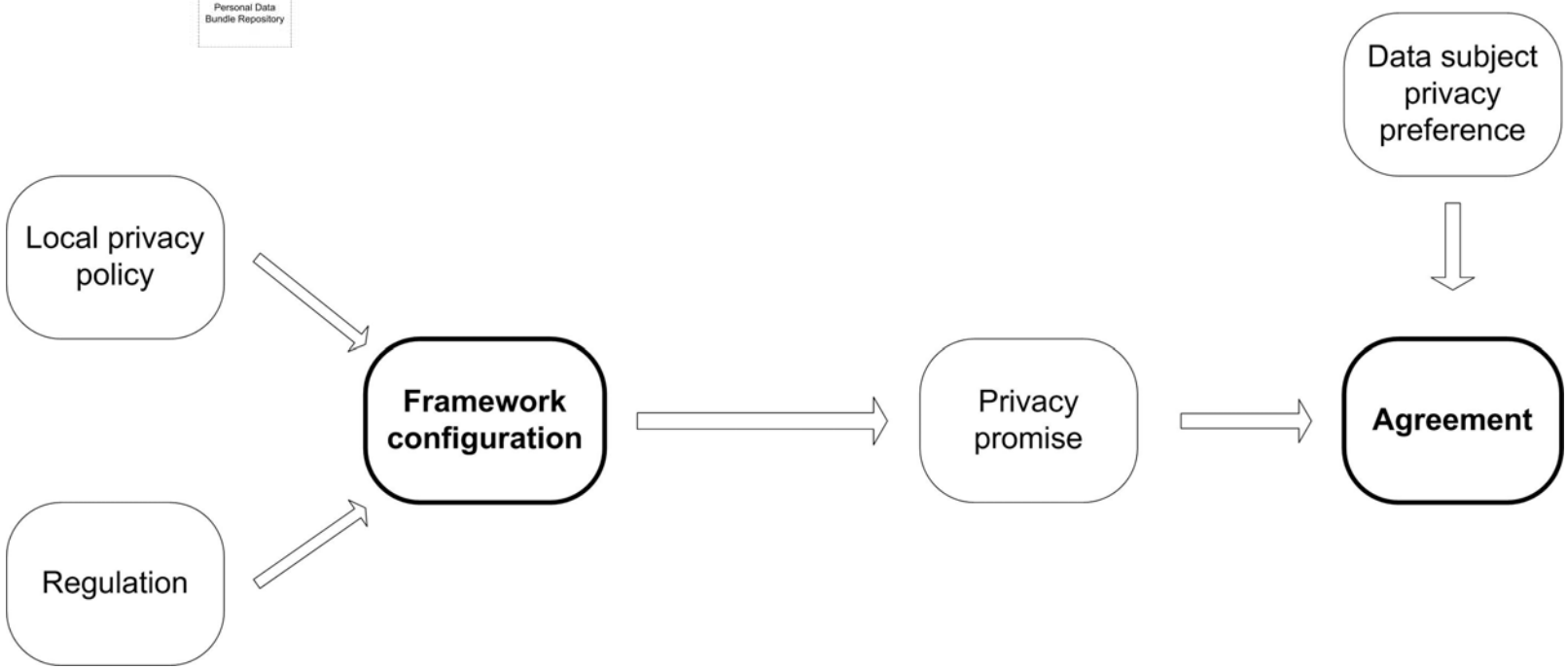
- Minimise the amount of personal data held
  - Minimise collection
  - Anonymisation
  - Pseudonymisation
- Enforce privacy policies
  - Proccesses and technology
  - Limit trust to data collector
  - Technology facilitates personalisation

Framework overview



Configuration





EPAL

- Enterprise Privacy Authorization Language
- Developed by IBM Research
- Submitted to W3C for standardisation (nov. 2003)
- EPAL elements are defined in XML

- EPAL specification:
  http://www.zurich.ibm.com/security/enterprise-privacy/epal/

EPAL policy

- A privacy policy written in EPAL is made up of privacy authorisation rules that:

  allow or deny actions on data-categories by user-categories for certain purposes under certain conditions while mandating certain obligations

EPAL vocabulary

- EPAL policies are based on a *vocabulary*, i.e. definitions of
  - Data categories
  - User categories
  - Actions
  - Purposes
  - Conditions
  - Obligations
- These must correspond to those used in the application or system

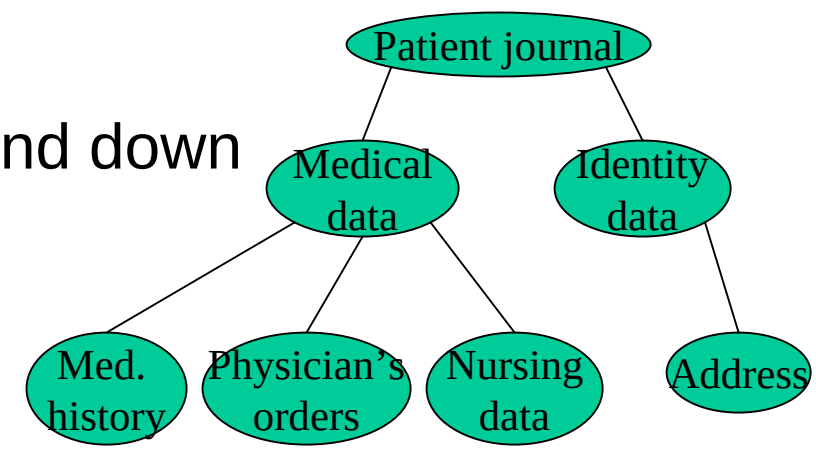
Interpretation of rules

- Rules are ordered by precedence
  - The first rule has highest precedence
  - Exceptions to a rule may be implemented by putting the exception first
- Rules are either ”allow” or ”deny” rules
- The rule interpretation algorithm may also output a list of obligations
- A default ruling of either ”allow” or ”deny” is defined to cover all cases not explicitly defined by rules

Hierarchies

- Data-categories, user-categories and purposes may be ordered in hierarchies (by specifying the ”parent”)

- ”Allow” rules inherit down
- ”Deny” rules inherit both up and down



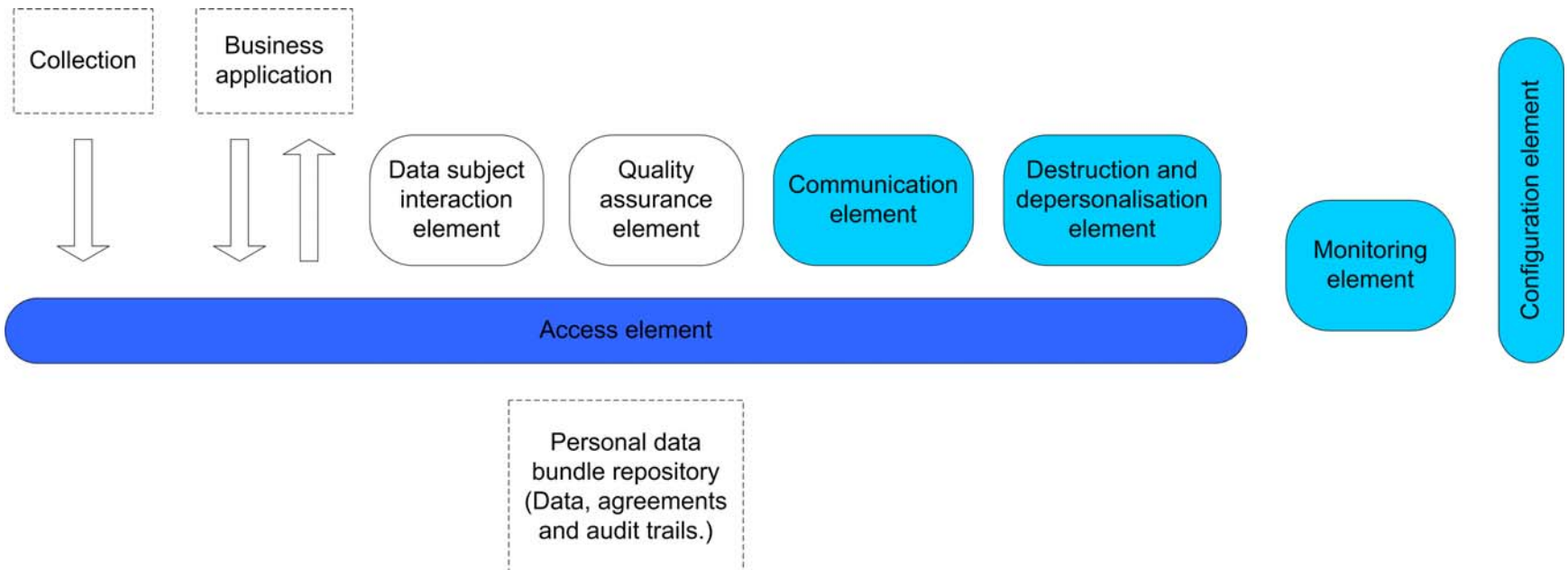
Rule examples

- General rule:
  - Deny everybody access to patient journal, personnel records or staffing schedules for any purpose.
- Exceptions:
  - Allow patient read access to patient journal for any purpose provided that patient ID matches patient journal ID.
  - Allow receptionists read and write access to patient identity data for the purpose of hospitalisation if the treating doctor has ordered it. Treating doctor must be alerted when patient has arrived.
  - Allow nurses read acces to physician's orders for the purpose of treatment.

Discussion

- Advantages:
  - Rules are relatively easy to define
  - It is possible to create very compact policies
  - No examples yet of rules that are impossible to specify
- Disadvantage?:
  - Not a formal language
- Challenges:
  - Agreeing on good vocabularies
  - Integration with application, in particular how to map procedures in an application on purposes
  - Obtaining context information for evaluation of conditions

Java Implementation



Functionalty

- Access control
  - Anonymisation
  - Pseudo domains and regulated linking between domains
- Logging
- Access recording
  - Record users access pattern/need
  - Recordings can for base for access policy
  - Recordings can be used for detecting anomalies

Privacy Access Control

- The purpose of the information access must coincide with the purpose stated when the information accessed was collected (purpose binding).
- Access could lead to obligations that must be addressed. For example, persons should be notified when someone run a credit check on them.
- Need for access control based on individual preferences.

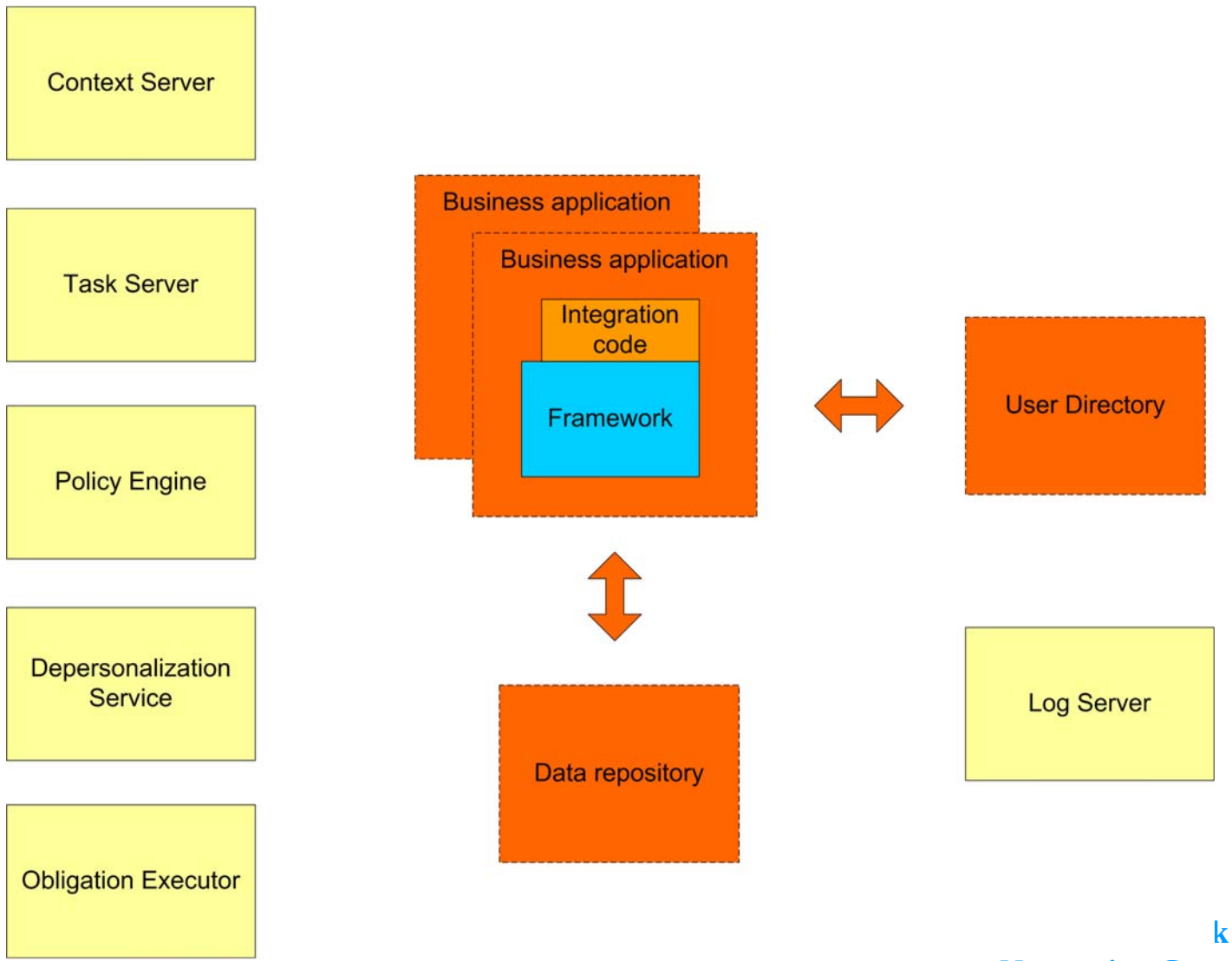
Information the framework must extract from the application

- The user who wants to access the data
- The action performed on the data
- The purpose (i.e. task) of the access
- Who is the data about (i.e. the data subject)?
- The type of data about the data subject that is accessed
- Information needed to evaluate conditions

Concepts

- Personal Data Objects
  - Objects that contain personal data
  - Have a *data subject* identified by an instance variable
- Data subject
  - A person that a set of personal data pertains to
- Privacy Meta Data
  - Meta data about personal data objects
- Task
  - Clients invoke a method while performing a particular *task*
  - Each task is performed with one and only one purpose

Overview


Context Server
Task Server
Policy Engine
Depersonalization Service
Obligation Executor
Business application
Business application
Integration code
Framework
User Directory
Data repository
Log Server

Using the Framework (Application Development)

- Design and implement application
  - Follow guidelines and conventions
- Develop integration code
  - GUI callbacks, user management and helper classes
- Annotate personal data classes
  - Type of instance variables, data subject and control points
  - Alternative: Annotate UML diagram
- Use tools to generate meta-data-configuration

Using the Framework (Application Deployment and Use)

- Choose control points
- Write vocabulary (e.g. in EPAL)
- Update generated meta-data-configuration
  - So that it is in compliance with vocabulary
- Configure services
  - Write task rules
  - Write local policy (e.g. in EPAL)
- Reach agreements with customers
  - Individual policies (e.g. in EPAL)

Annotations example

```
public class PatientJournal implements IPatientJournal {

    @no.nr.privacy.framework.integration.DataSubjectHelper
    ("no.nr.epr.privacyintegration.Helper")
    @no.nr.privacy.framework.integration.PersonalDataType("ID")
    public int id;

    @no.nr.privacy.framework.integration.PersonalDataType
    ("LOCATION")
    public int roomnumber;

    @no.nr.privacy.framework.integration.ControlPoint
    (type="ACCESS", variant="PDOConstructorAccess")
    public PatientJournal(...) {}

    @no.nr.privacy.framework.integration.ControlPoint
    (type="ACCESS", variant="PDOVOutputAccess",
    variables={"roomnumber"})
     public int getRoomnumber() {
         return roomnumber;
     }
```

Access Control Points

- Control access to methods that create and populate personal data objects.
- Control access to methods that retrieve personal data objects.
- Control access to methods that read or write to personal data object's instance variables.
- Control accesses to other types of methods that take personal data as input or access personal data objects

Thank you!

Ragni.Ryvold.Arnesen@nr.no
Jerker.Danielsson@nr.no
Bjørn.Nordlund@nr.no

http://snipsnap.nr.no:8668/privacy/space/start