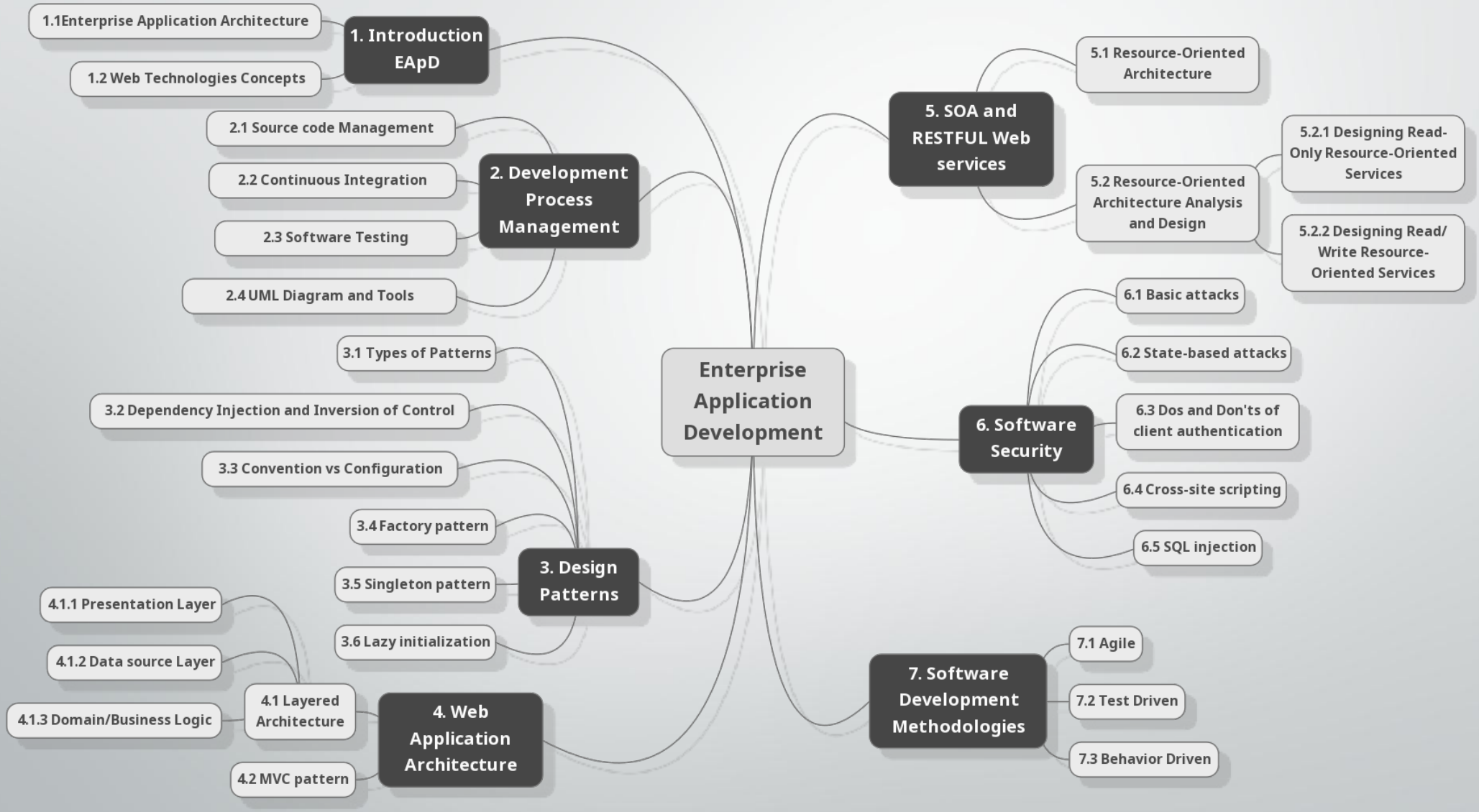
Enterprise Application Development

[BE SE-7th Semester]

Nepal College of Information Technology

POKHARA UNIVERSITY

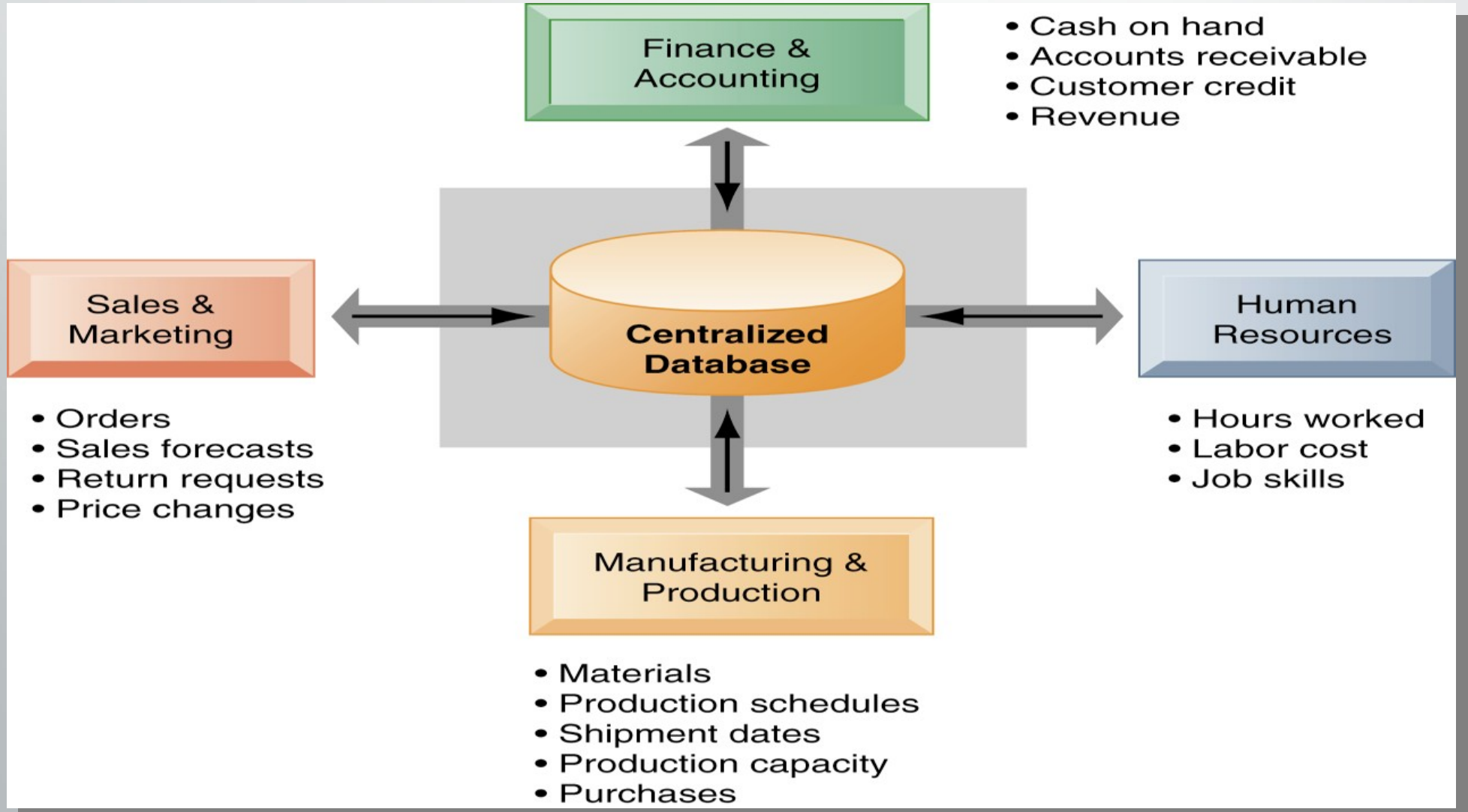
Enterprise Application Development



Enterprise

- An Organizational unit- from a department to a whole corporation
- Working together to achieve some common goals.
- Organization come in all shape and sizes,
- large and small, for profit and non profit,
- governmental and non governmental

How enterprise System Works?



Enterprise

- Common needs:
  - Information sharing and processing
  - Assets management and tracking
  - Resource planning
  - Customer or client management,
  - Protection of business knowledge and so on
- Enterprise software is used to collectively refers to all software involved in supporting those common elements of an enterprise.

Architecture

- It is a formal description of an enterprise,
- a detailed map of the enterprise
- at component level to guide its implementation.
- The structure of components, their inter-relationships and the principles and guidelines governing their design and evolution over time.

Enterprise Architecture Frameworks

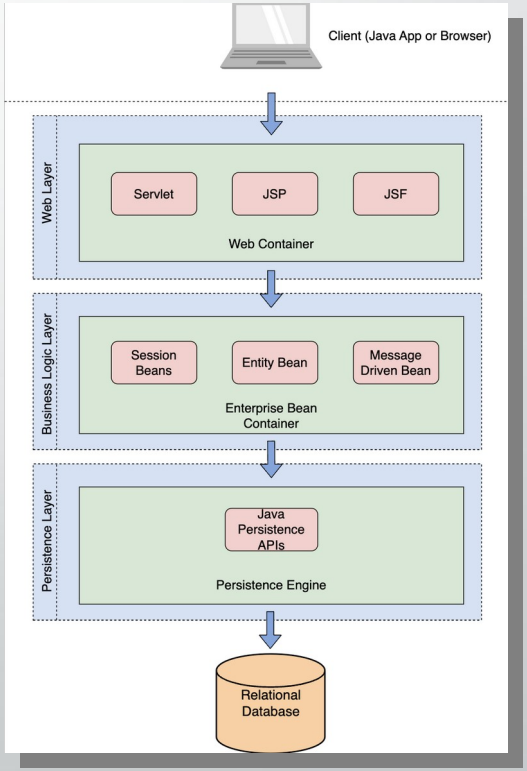
- Frameworks help people organize and assess completeness of integrated models of their enterprises.
- An Architectural Framework gives a skeletal structure that defines suggested architectural artifacts,
- describes how those artifacts are related to each other, and provides generic definitions for what those artifacts might look like.
- EAF says:

  How to create and use an enterprise architecture.

  Principles and practices for creating and using architectural description of the

Purpose of Framework

- Organize integrated models of an enterprise
- Assess completeness of the descriptive representation of an enterprise
- Understand an organization or a system
- Assist in identification and categorization
- Provide a communication mechanism
- Help manage complexity
- Identify the flow of money in the enterprise

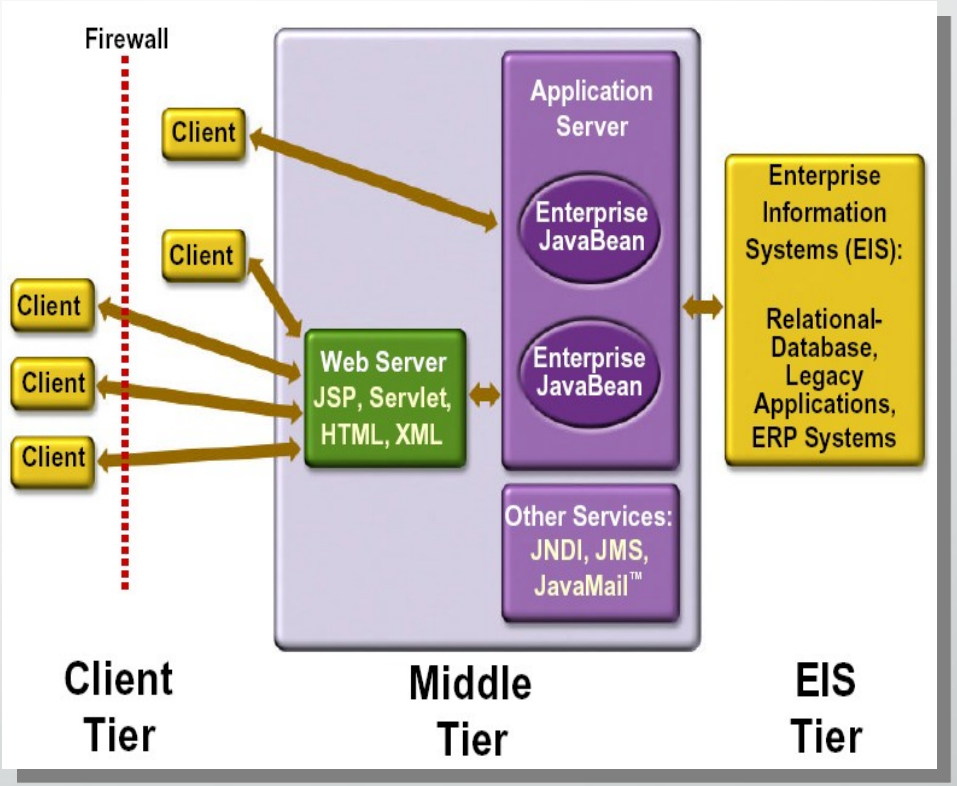


Platform for Enterprise Solutions

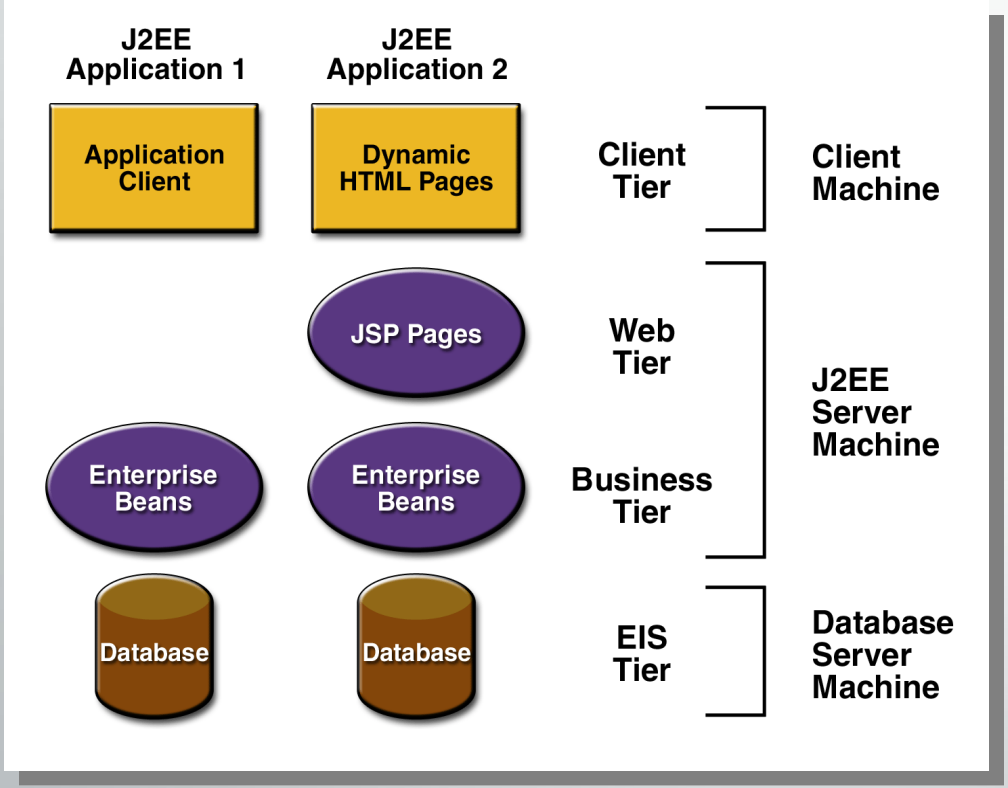
- Jakarta EE, formerly Java Platform, Enterprise Edition and Java 2 Platform, Enterprise Edition, is a set of specifications, extending Java SE with specifications for enterprise features such as distributed computing and web services
- The J2EE platform uses a multi-tiered distributed application model for both enterprise applications
- Application logic is divided into “components” according to function, and the various application components
- that make up a J2EE application are installed on different machines depending on the tier in the multi-tiered J2EE environment to which the application component belongs

J2EE Architecture

Three-tiered applications that run in this way extend the standard two-tiered client and server model by placing a multithreaded application server between the client application and back-end storage

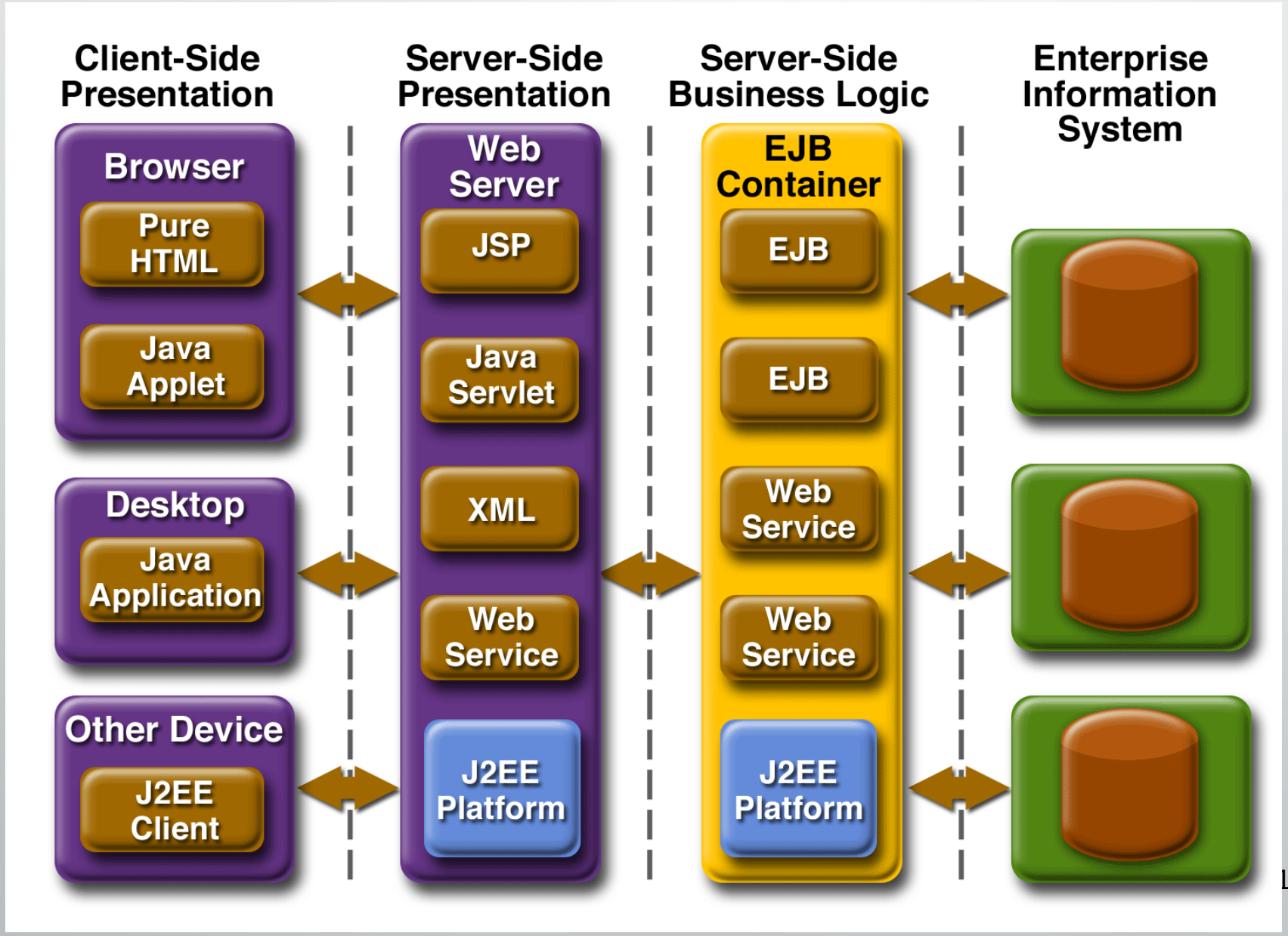


J2EE Architecture



- J2EE multi-tiered applications are generally considered to be three-tiered applications because they are distributed over three different locations
  - client machines
  - the J2EE server machine
  - the database or legacy machines at the back end

Multi-tier Architecture of Enterprise Java



J2EE Containers

- The application server maintains control and provides services through an interface or framework known as a container
- There are five defined container types in the J2EE specification
- Three of these are server-side containers:
  - The server itself, which provides the J2EE runtime environment and the other two containers
  - An EJB container to manage EJB components
  - A Web container to manage servlets and JSP pages

Client J2EE Containers

- The other two container types are client-side:
  - An application container for stand-alone GUIs, console
  - An applet container, meaning a browser, usually with the Java Plug-in

J2EE Components

- As said earlier, J2EE applications are made up of components
- A *J2EE component* is a self-contained functional software unit that is assembled into a J2EE application with its related classes and files and that communicates with other components

Components

- Client components run on the client machine, which correlate to the client containers
- Web components -servlets and JSP pages
- EJB Components

Packaging Applications and Components

- Under J2EE, applications and components reside in Java Archive (JAR) files
- These JARs are named with different extensions to denote their purpose, and the terminology is important

Various File types

- Enterprise Archive (EAR) files represent the application, and contain all other server-side component archives that comprise the application
- Client interface files and EJB components reside in JAR files
- Web components reside in Web Archive (WAR) files

Deployment Descriptors

- Deployment descriptors are included in the JARs, along with component-related resources
- Deployment descriptors are XML documents that describe configuration and other deployment settings (remember that the J2EE application server controls many functional aspects of the services it provides)
- The statements in the deployment descriptor are declarative instructions to the J2EE container; for example, transactional settings are defined in the deployment descriptor and implemented by the J2EE container

Deployment Descriptors

- Most J2EE Web Services vendors provide a GUI tool for generating deployment descriptors and performing deployment because creating manual entries is tedious and error prone
- The deployment descriptor for an EJB component must be named ejb-jar.xml, and it resides in the META-INF directory inside the EJB JAR file

EJB Components

- EJB components are server-side, modular, and reusable, comprising specific units of functionality
- They are similar to the Java classes we create every day, but are subject to special restrictions and must provide specific interfaces for container and client use and access
- We should consider using EJB components for applications that require scalability, transactional processing, or availability to multiple client types

EJB Components- Major Types

- **Session beans**
  - These may be either *stateful* or *stateless* and are primarily used to encapsulate business logic, carry out tasks on behalf of a client, and act as controllers or managers for other beans
- **Entity beans**
  - Entity beans represent persistent objects or business concepts that exist beyond a specific application's lifetime; they are typically stored in a relational database

The *home* and *component* interface

- A bean's home interface specifies methods that allow the client to create, remove, and find objects of the same type
- The home interface provides bean management and life cycle methods
- EJB functionality is obtained through the bean's component interface, which defines the business methods visible to, and callable by, the client
- The developer writes the component interface, and the container creates the implementation for client interaction

Enterprise Java Beans (EJB)

- Enterprise JavaBeans is a specification for creating server-side secure, scalable, transactional, multi-user secure enterprise-level applications.
- These server-side components, called enterprise beans, are distributed objects that are hosted in Enterprise Java Bean containers and provide remote services for clients distributed throughout the network.

Java Beans vs. EJB

- Can be either visible non-visible.
- Local Invocation
- Synchronous Invocation

- Decidedly non-visible remote objects
- Remote and Local Invocation
- Synchronous and Asynchronous Invocation
- Object Pooling
- Transparent Persistence
- Supports Transactions
- Support Relationships between entity EJBs
- J2EE Security Features

Advantages of EJB

- Simplifies the development of middleware components that are secure, transactional, scalable & portable.
- Simplifies the process to focus mainly on business logic rather than application development.
- Overall increase in developer productivity
- Reduces the time to market for mission critical applications

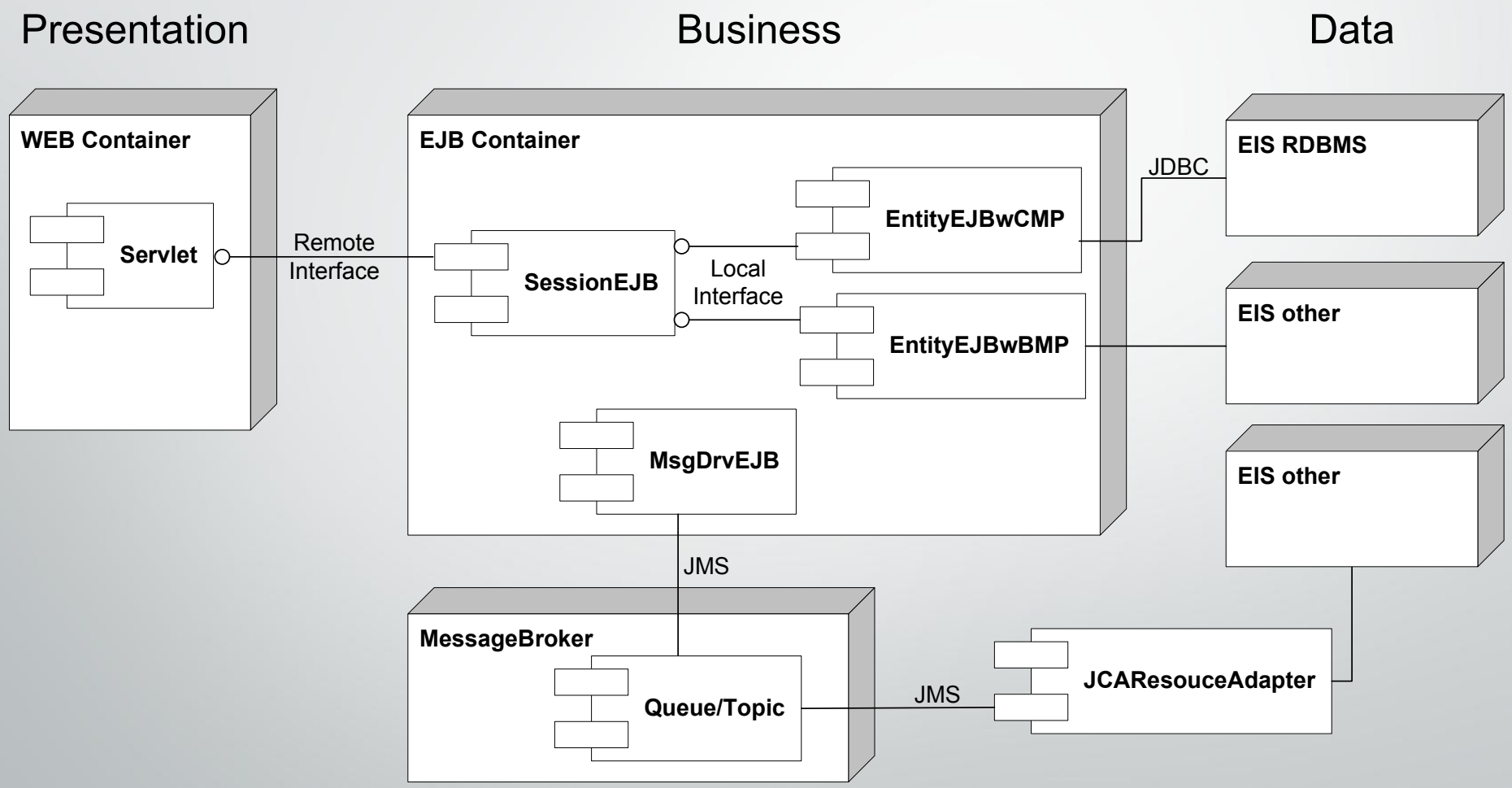
Purpose of EJBs

- SESSION Beans (verbs of the system):
  - Model task or workflow
  - Façade for Entity beans
  - Maintain conversational state with clients
- ENTITY Beans (nouns of the system):
  - Object/Relational (O/R) mapping
  - Transparent and implicit persistence with transaction support
- Message Driven Beans:
  - Asynchronous communication with Message-Oriented Middleware
  - Conduit for non-J2EE resources to access Session and Entity Beans via Java™ EE Connector Architecture (JCA) Resource adapters.

EJB server
aka Enterprise Java Server (EJS)

- EJS are analogous to the CORBA ORB.
- Part of an application server that hosts EJB containers
- EJBs do not interact directly with the EJB server
- EJB specification outlines eight services that must be provided by an EJB server:
  - Naming
  - Transaction
  - Security
  - Persistence
  - Concurrency
  - Life cycle
  - Messaging
  - Timer

Three Tier Architecture Using EJBs

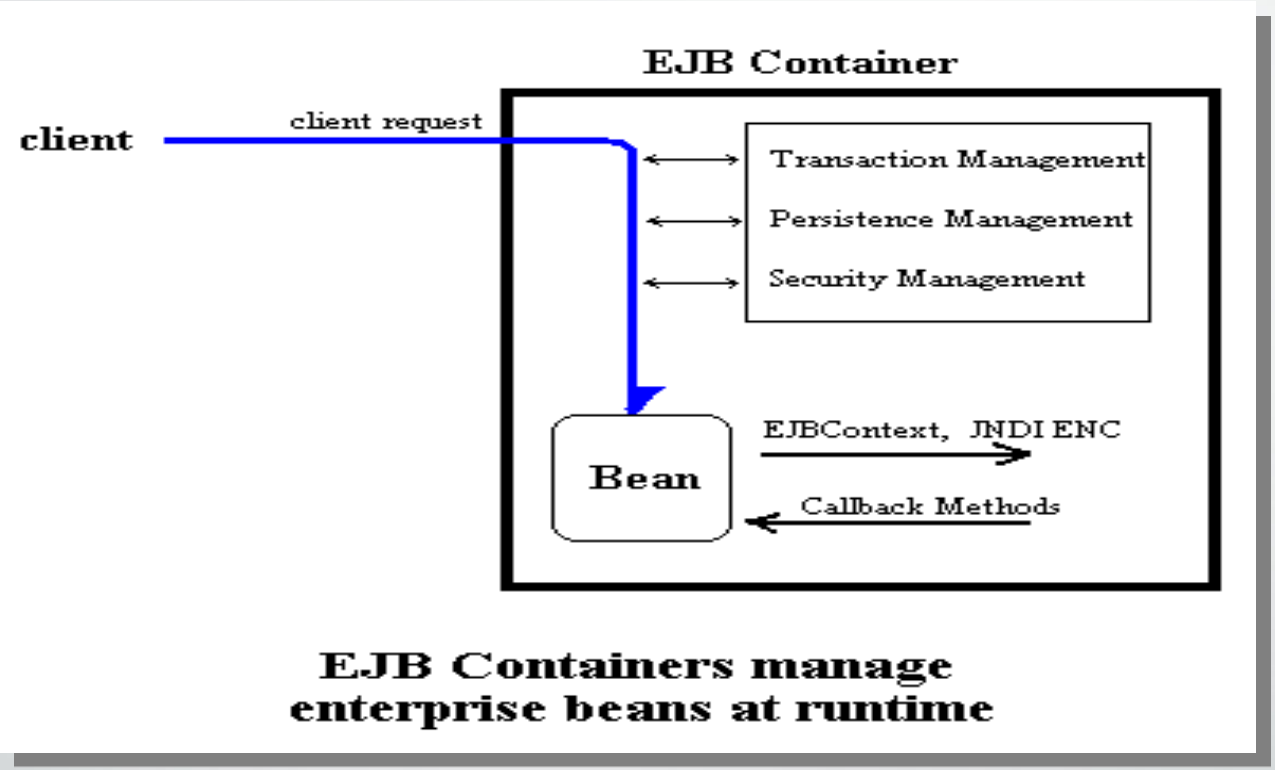


EntityEJBwCMP = Entity Bean with Container Managed Persistence
EntityEJBwBMP = Entity Bean with Bean Managed Persistence
MsgDrvEJB = Message Driven EJB

EJB Container

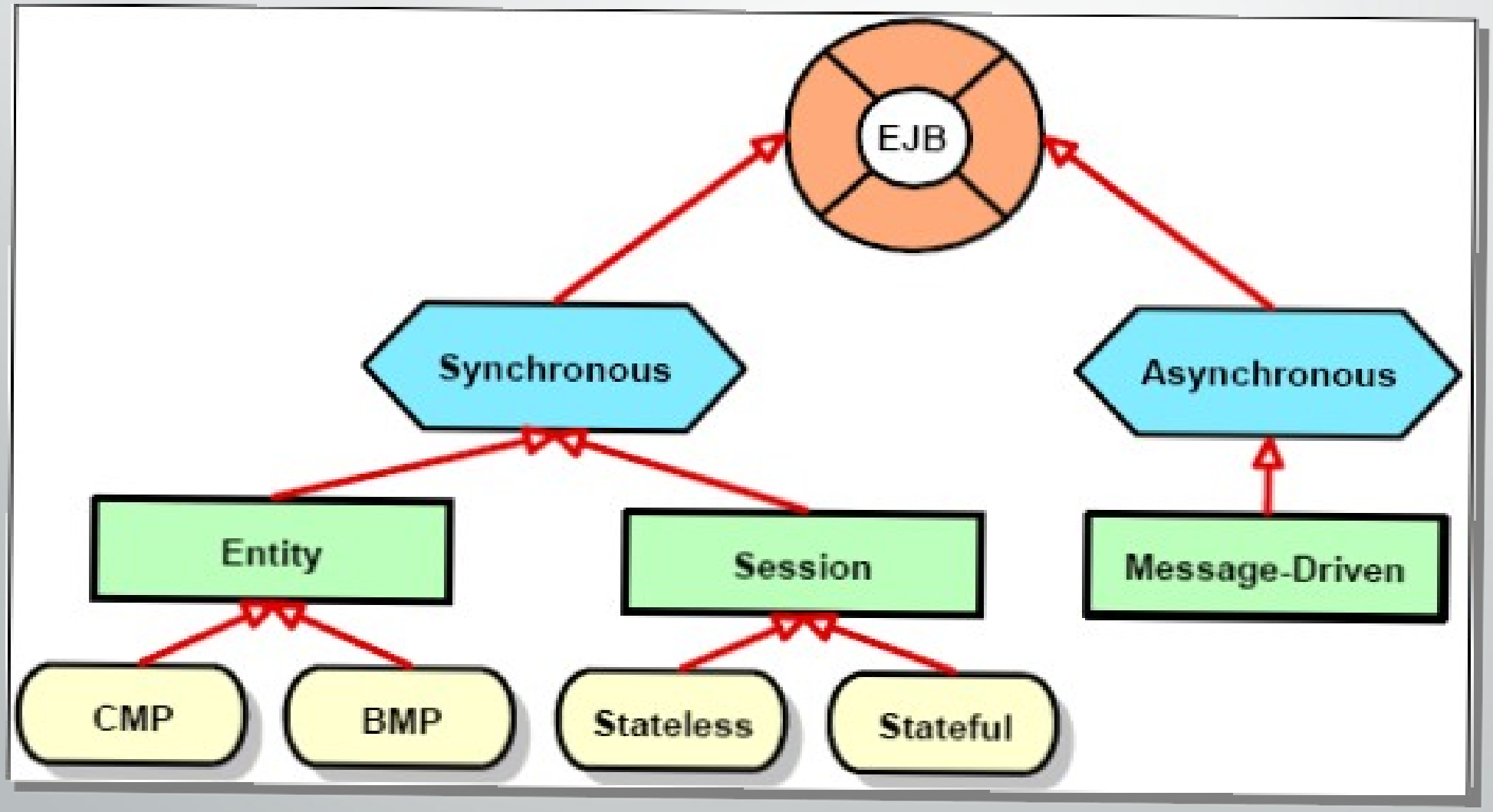
- Functions as a runtime environment for EJB components beans
- Containers are transparent to the client in that there is no client API to manipulate the container
- Container provides EJB instance life cycle management and EJB instance identification.
- Manages the connections to the enterprise information systems (EISs)

EJB Container



- Finds EJB container via JNDI.
- Invokes methods on EJB beans.

EJB components



Entity EJB

- CMP (Container Managed Persistence)
  - Container maintains persistence transparently using JDBC calls
- BMP (Bean Managed Persistence)
  - Programmer provides persistence logic
  - Used to connect to non-JDBC data sources like LDAP, mainframe etc.
  - Useful for executing stored procedures that return result sets

Zachman Enterprise Architecture Framework

- The Zachman Framework is an EAF which provides a formal and highly structured way of viewing and defining an enterprise.
- It consists of a two dimensional classification matrix
- Six communication questions and Five levels of reification
- abstract ideas (Scope level) into concrete ideas (Operations level)
- The Zachman Framework is a schema for organizing architectural artifacts
- design documents, specifications, and models) that takes into account both whom the artifact targets (for example, business owner and builder) and what particular issue (for example, data and functionality) is being addressed.
- a methodology in that it does not imply any specific method or process for collecting, managing, information that it describes

Zachman Framework

	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*
Objective/Scope (contextual) *Role: Planner*	List of things important in the business	List of Business Processes	List of Business Locations	List of important Organizations	List of Events	List of Business Goal & Strategies
Enterprise Model (conceptual) *Role: Owner*	Conceptual Data/ Object Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
System Model (logical) *Role:Designer*	Logical Data Model	System Architecture Model	Distributed Systems Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
Technology Model (physical) *Role:Builder*	Physical Data/Class Model	Technology Design Model	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
Detailed Reprentation (out of context) *Role: Programmer*	Data Definition	Program	Network Architecture	Security Architecture	Timing Definition	Rule Speculation
Functioning Enterprise *Role: User*	Usable Data	Working Function	Usable Network	Functioning Organization	Implemented Schedule	Working Strategy

Government Enterprise Architecture Frameworks (GEAFs)

- A coordinated set of activity areas involving one or more public organizations and possibly third party entities from private organizations or civil society,
- an EA provides technical descriptions of the organizational goals, business and administrative processes, information requirements, supporting applications and technology infrastructure of the enterprise.
- These descriptions are typically captured in the form of models, diagrams, narratives, etc.
- A Government Enterprise Architecture (GEA) may be associated with a single agency or span functional areas transcending several organizational boundaries, e.g. health care, financial management and social welfare.

Why GEAFs?

- Government Enterprise Architecture is a guided by the architectural framework which provides guidelines, architecture principles, architecture development methodology, content meta model and the reference model from business and technology services perspective defining the principles of interoperability between departments for better and efficient service delivery to the citizens and businesses in a country.
- Understanding, clarifying and optimizing the inter-dependencies and relationships among business operations, the underlying IT infrastructure and applications that support these operations in government agencies and in the context of specific government enterprises
- Establishing a basis for agencies to share information, knowledge and technology and other resources or jointly participate in the execution of business processes
- Optimizing ICT investment and business cases across the whole of government by enabling the opportunities for collaboration and sharing of assets, thus reducing the tendency for duplicated and poorly integrated IT resources and capabilities

Aims of GEAFs

- The project aims to provide policy guidelines for the development of Government Enterprise Architecture Frameworks (GEAFs),
- Establish concrete requirements for such a framework in Macao, and
- Provide recommendations on how elements of a Macao GEAF (MGEAF) could be built from existing Government EA Frameworks, Reference Models, Methods, and Modeling Framework.
- The project will also provide an example of agency-specific EA based on the recommended Macao framework.

Objectives:

1) Improving understanding and contributing to the body of knowledge of GEA through foundational research

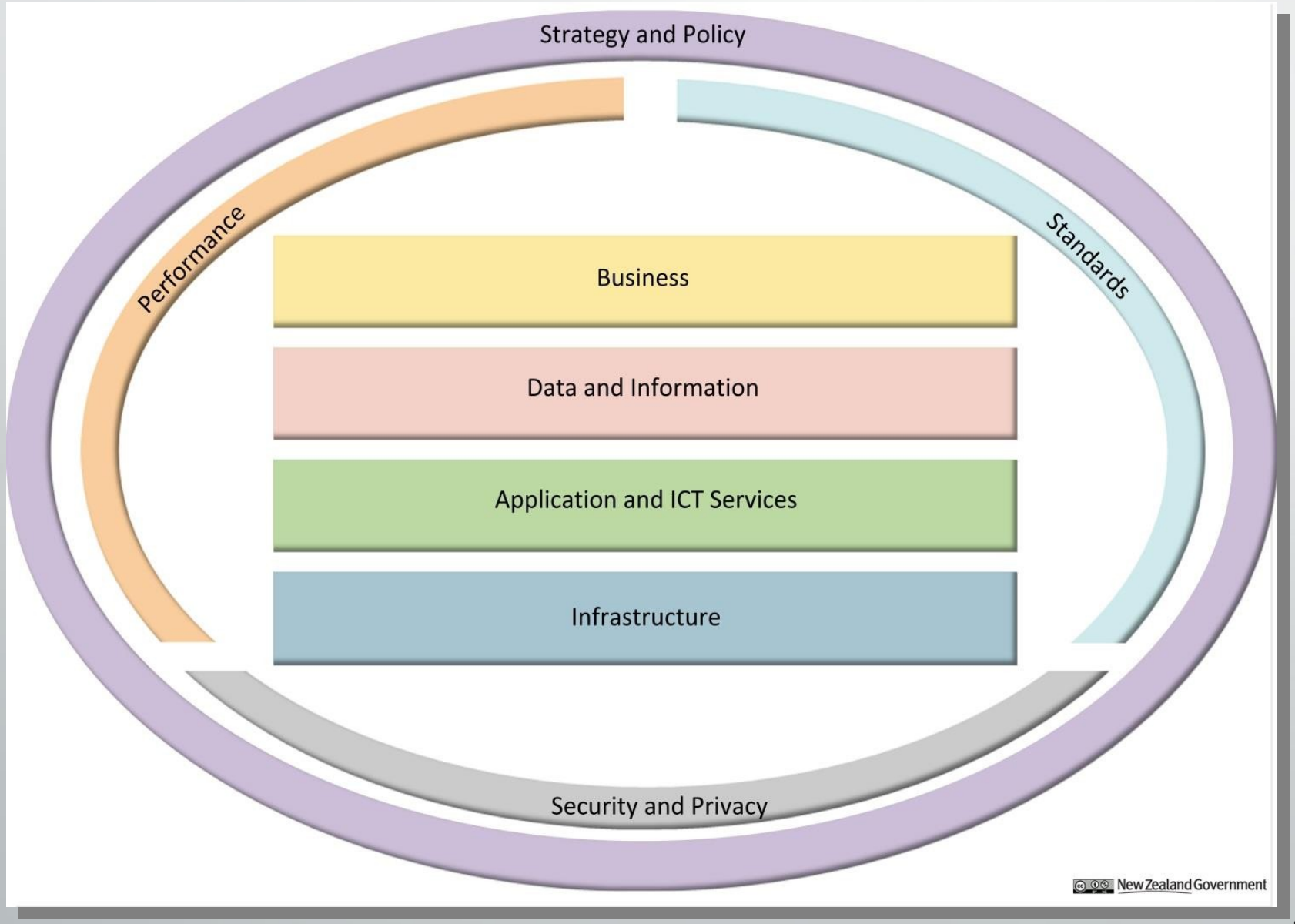
2) Enhancing EA practice by providing policy guidelines and development of Government EA Frameworks based on results from (1) with the supporting toolkit

3) Understanding the factors that contribute to wide adoption of EA practice within a government

4) Building capacities of government agencies and their architects through development of courseware for educational and training purposes as well as the use of tools in (2)

5) Dissemination of project output (1 through 4) through various channels including publications (books, journals, conference papers and technical reports), schools and courses, seminars, workshops and projects.

GEAF - NewZealand



Primary Outcomes

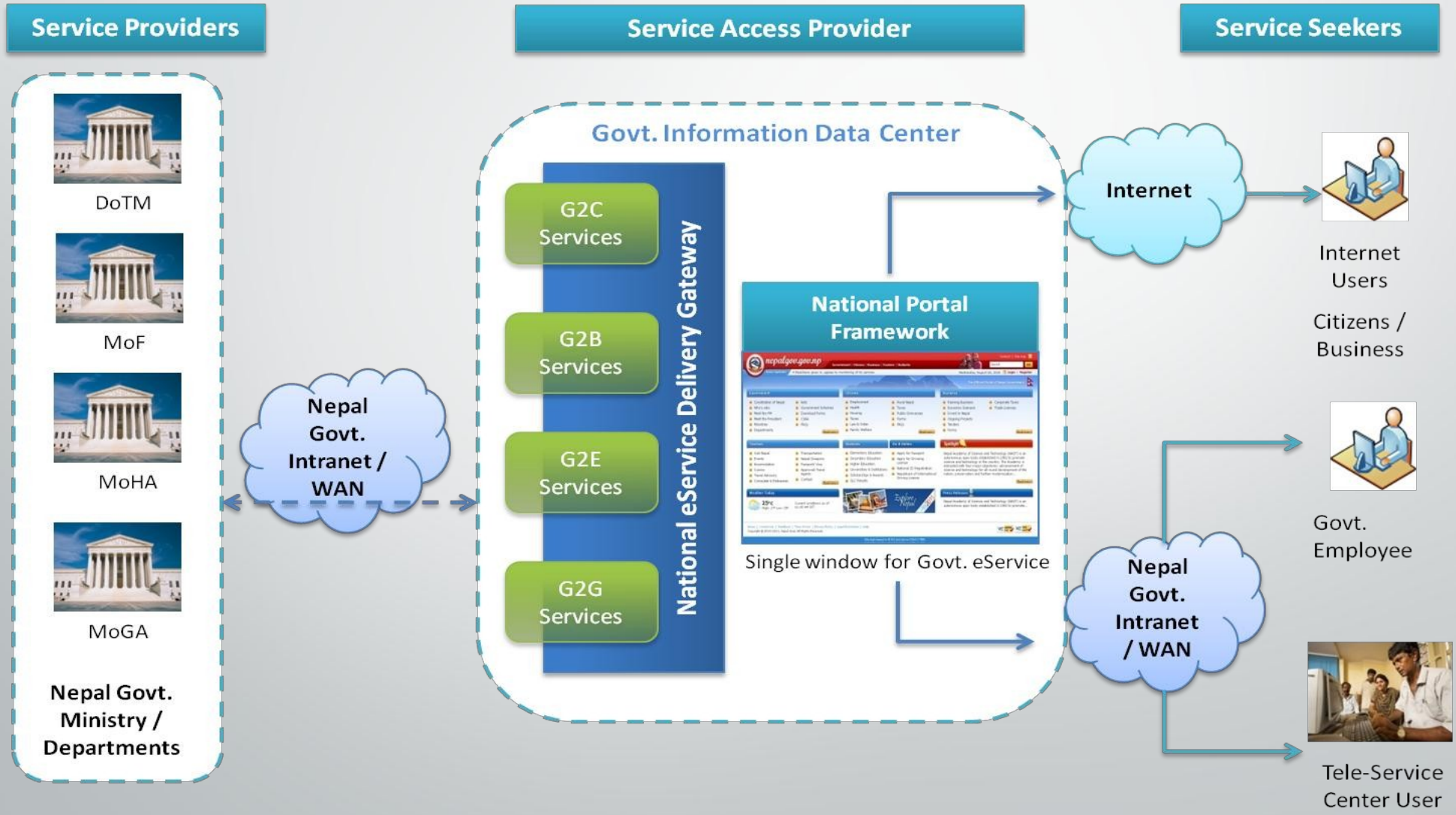
The GEA-NZ v3.0 helps in enabling the following four primary outcomes:

- **Success of Government Goals and Objectives**
  Provide a consistent view and accurate information within and across agencies to support planning and decision making
- **Functional Integration**
  Facilitate and encourage interoperability within and across agencies and between programs and enhanced services by the use of Enterprise Architecture standards
- **Authoritative Reference**
  Provide an integrated, consistent view of strategic goals, business services and enabling technologies across the entire organization, including programs, services, and systems
- **Resource Optimization**
  Provide a harmonized and consistent view of all types of resources in each functional area, program, and system area

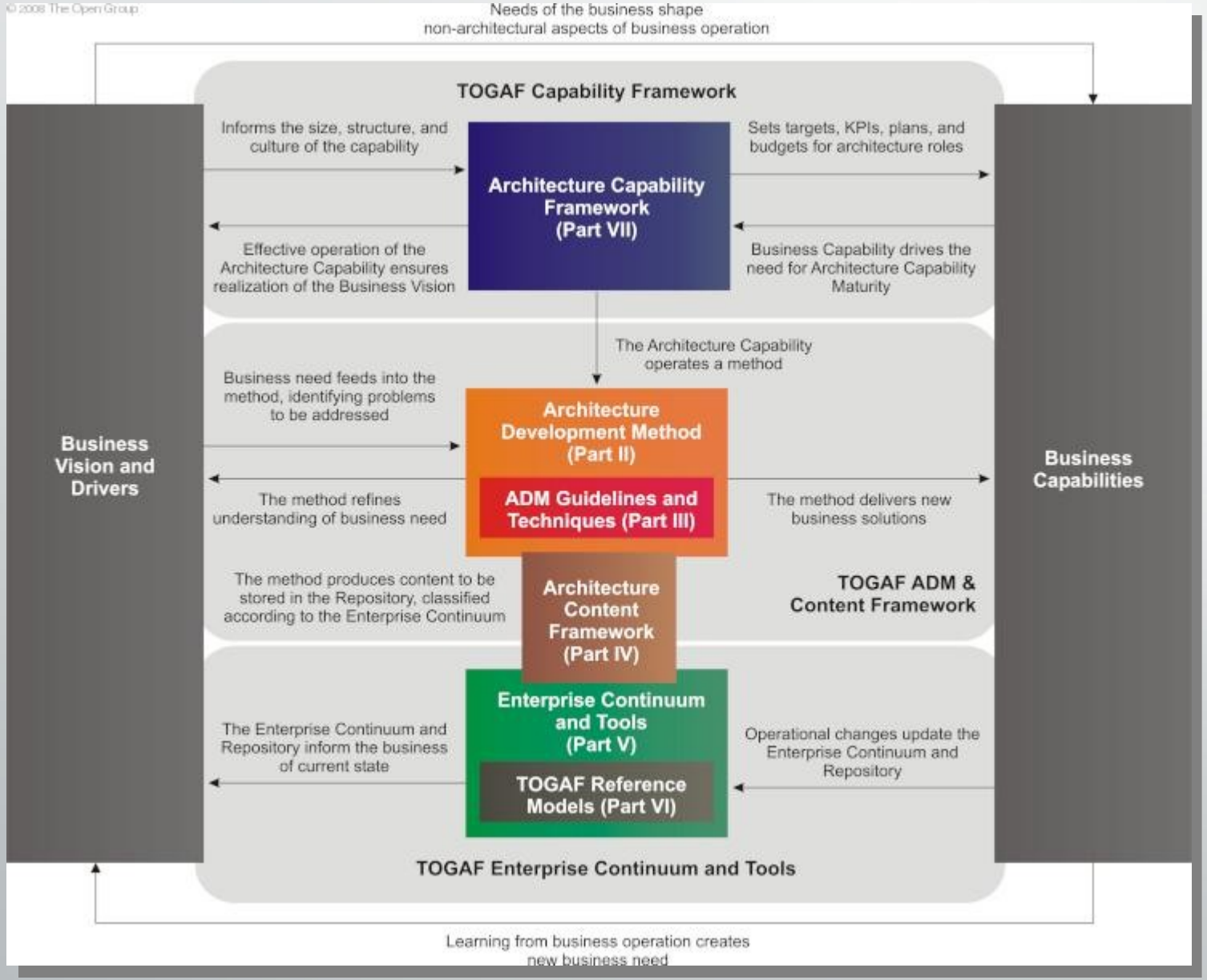
Dimensions

Dimension	Description
Strategy and Policy	The GEA-NZ Strategy and Policy reference model is des between the architecture and strategic goals, policies a
Performance	The GEA-NZ Performance reference model describes pe related metrics that apply across other dimensions of G
Business	A Business reference model is a generic representation products and services that deliver the outputs of the or the Business reference model emphasises aspects of cu channel shift that are objectives of the ICT strategy and
Data and Information	GEA-NZ Data and Information primary purpose is to dis share and reuse information within and across agencies and artefacts that can be generated from the data arch data and information governance framework, and matu
Application and ICT Services	GEA-NZ Application and ICT Services describes the busi 'Software as a Service', that support the business proce includes core business applications, COTS corporate ap computing applications.
Infrastructure	GEA-NZ Infrastructure describes the technology infrastr application and business processes of the enterprise. It outsourced or cloud capabilities.
Security and Privacy	GEA-NZ Security and Privacy does not prescribe a "new an enterprise architecture context to the relevant secu for government agencies, and other guideline artefacts (or life-cycle) for ICT security and privacy.
Standards	GEA-NZ Standards categorises the information and tech the NZ government. The existing standards base incorp Interoperability Framework (eGIF) standards is structur Government enterprise architecture will restructure th the GEA-NZ v3.0 dimensions i.e. Business, Data and Inf

Nepal GEA Service Delivery Landscape



Nepal Government EAF



Nepal Government Enterprise Architecture Framework

- PwC had followed the industry standard TOGAF for developing the government enterprise architecture for Govt. of Nepal.
- Using TOGAF as the architecture framework will allow architectures to be developed that are consistent, reflect the needs of stakeholders, employ best practice, and give due consideration both to current requirements and to the likely future needs of the government.
- TOGAF provides a platform for adding value, and enables users to build genuinely open systems-based solutions to address their business issues and needs.
- Besides it underpins a practical standardized methodology of implementing successful EA to organizations and is widely accepted and the most adopted architectural framework.
- The Open Group Architecture Framework (TOGAF 9.0) was adopted for the development of the Nepal GEA framework.
- TOGAF was tailored as appropriate to meet the needs of the Govt. of Nepal‘s EA requirements.

Enterprise Application Development