

- Focus on building an organization culture to "be agile" rather than "follow Agile".
- Software agility delivered through team agility
- Team agility delivered through coaching teams to be self-organized
- Self-organized teams using agile tools, techniques and technologies.

Agility Services

- Software Craftsmanship
- Midas Touch - Agility in software maintenance
- Agile Enterprise Architecture solutions
- Agility Nurseries (Agile ODC)

Organization Metamorphosis

- Agility Assessment Radars and Roadmap
 - Team Agility Assessment
 - Value Stream Mapping
 - Shared Vision and Team Chartering
- Team Agility Coaching, Executive Coaching
 - Scrum Coaching
 - XP Engineering Practices Coaching
 - Lean Software Development Coaching



Ajay Danait

Vice President - Agile Global Strategies

Stixis Technologies Pvt.Ltd.

AGR Plaza, L-150, 5th Main,6th Sector

HSR Layout, Bangalore 560102, India

Tel: +91 80 43398181 Cell: +91 99025 43808

Email: ajay.danait@stixis.com

www.stixis.com



Agile Architecture Retrospective

An Inspect-N-Adapt Tool For the Enterprise

Audience Expectations

Primary Scope

Building a motivated, self-organizing architecture team

- Motivated Individuals
- Self – Organizing Teams
- Architect Persona

Retrospective on architecture

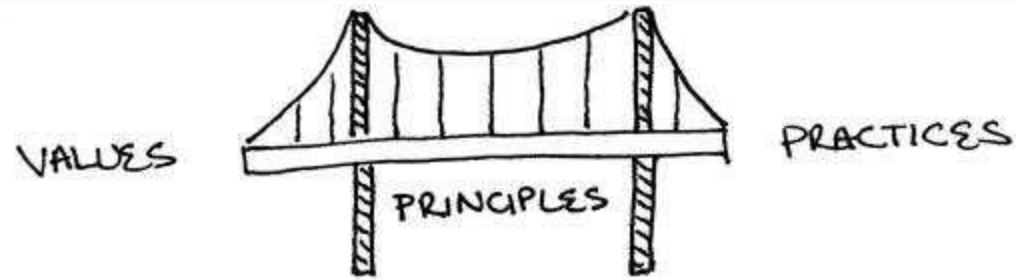
Secondary Scope

Software Craftsmanship and Architecture

Introducing Agility in Enterprise Architecture

Context Setting

Values, Principles and Practices : Relationship



Values bring purpose to Practices, Practices are evidence of Values, Practices bring accountability to Values.

Bridging the gap between Values and Practices are Principles. Principles are context specific guidelines.

e.g.

practice → Pair Programming

values → “communication” and “feedback”

principle → driver-navigator principle – dual thinking hats of constructing and preventing from breaking.

practice → Companion Planting used in agriculture

values → “sustainability” and “responsibility”

principle → Diversity is nature's design, cooperation is more apparent than competition in plants, crop stability tends to increase with increasing diversity.

Sources: -

Extreme Programming – Embrace Change by Kent Beck

Intercropping Principles and Production Practices by National Sustainable Agriculture Service

Discuss with your partner

An instance of mapping a **practice** to its underlying
values and **principles**

Values, Principles and Practices Of A Manifesto

man·i·fes·to  (măñ'ə-fēs'tō)

n. pl. man·i·fes·toes or man·i·fes·tos

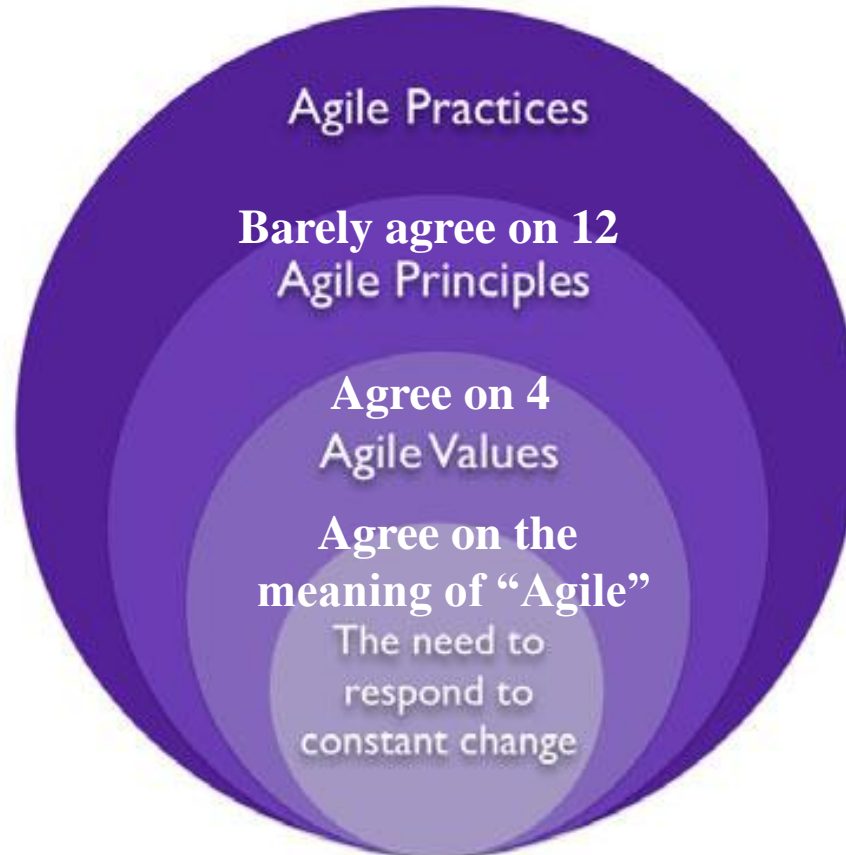
A public declaration of principles, policies, or intentions
(based on a commonly agreed set of values)

Source: - TheFreeDictionary.org

The Agile Manifesto

Layers of the Agile Manifesto

**Agree to disagree on
detailed project specific
ground tactics and prescriptive**



Individuals and Interactions (amongst Individuals)

Values

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Principles behind the Agile Manifesto Value - Individuals and Interactions

We follow these principles:

Build projects around **motivated individuals**. Give them the environment and support they need, and trust them to get the job done.

The best architectures, requirements, and designs emerge from **self-organizing teams**.

At regular intervals, the team **reflects** on how to become more effective, then **tunes and adjusts** its behavior accordingly.

Discuss with your partner

1. An Instance of Creative Cycle practice
2. An Instance of Survival Cycle practice

That maps to the **motivated individuals** principle

Principles behind the Agile Manifesto Value - Individuals and Interactions

We follow these principles:

Build projects around **motivated individuals**. Give them the environment and support they need, and trust them to get the job done.

The best architectures, requirements, and designs emerge from **self-organizing teams**.

At regular intervals, the team **reflects** on how to become more effective, then **tunes and adjusts** its behavior accordingly.

Architecture Manifesto(s)

SOA Manifesto

Service orientation is a paradigm that frames what you do.
Service-oriented architecture (SOA) is a type of architecture
that results from applying service orientation.

We have been applying service orientation to help organizations
consistently deliver sustainable business value, with increased agility
and cost effectiveness, in line with changing business needs.

Through our work we have come to prioritize:

Business value over technical strategy

Strategic goals over project-specific benefits

Intrinsic interoperability over custom integration

Shared services over specific-purpose implementations

Flexibility over optimization

Evolutionary refinement over pursuit of initial perfection

That is, while we value the items on the right, we value the items on the left more.

Probably A Software Architect Manifesto Candidate

- Accounting cost-of-change in upfront decisions **over** documenting architecture
- Live real-time documentation **over** comprehensive encyclopedias and glossaries
- Architecting non-functional requirements early **over** optimizing at the end
- Evaluating technology to business needs **over** latest buzzwords solution
(Just because it looks good on your resume / CV)
- Exploratory pointers / solutions **over** "It depends" consultant syndrome
- Experimentation **over** freezing on design early in product lifecycle
- Executable skeleton code **over** discussions, meetings and diagrams
- Team collaboration **over** solo architecture responsibilities
- Collective accountability **over** responsibility delegation to implementation team
- Coaching team members **over** individual knowledge consolidation
- Pragmatism **over** perfection
- Real problems **over** intellectual (self) stimulation
- Simplicity and common sense **over** complexity in ideas

Reference: - CodingTheArchitecture.com (Simon Brown)

Definition Of An Architect

Discussion

Being An Architect

“Crown” OR “Cap”

Title OR Role

Craftsperson
~~Craftsmen~~
Agile ~~Architects~~

Manifesto for Software Craftsmanship

Raising the bar.

As aspiring Software Craftsmen we are raising the bar of professional software development by practicing it and helping others learn the craft. Through this work we have come to value:

Not only working software,
but also **well-crafted software**

Not only responding to change,
but also **steadily adding value**

Not only individuals and interactions,
but also **a community of professionals**

Not only customer collaboration,
but also **productive partnerships**

That is, in pursuit of the items on the left we have found the items on the right to be indispensable.

Architect to Craftsperson

What is Software Craftsmanship?

“Software Craftsmanship is all about putting responsibility and pride back into the software development process.”

“The best processes in the world will not save a project from failure if the people involved do not have the necessary skills to execute the process; conversely, really good developers can make any process work”

“A Software Craftsman is a continuous learner. When he doesn’t work, he spends his time studying, to find new methods and tools can refine him as a Software Craftsman”

- Pete McBreen, Software Craftsmanship: The New Imperative

Software Craftsmanship is about

- Taking responsibility
- Taking pride in work
- “Signing” your work
- Being a continuous learner
- Practicing deliberately
- Writing code
- Having the right attitude
- Contributing to the community

Architect to Craftsperson through Apprenticeship

How should I become an expert in software craftsmanship?

- Read and understand the concepts on Apprenticeship Patterns
 - David Hoover, Adewale Oshineye
- Find a mentor
- Study, Train and Practice
 - Performing Code Katas
 - Performing Coding Dojos
 - Performing Acceptance-Test based
 - Learning TDD
 - Learning programming paradigms – functional, dynamic, statically typed languages
 - Refactoring – keep your code healthy
 - Learning design patterns, tools and frameworks
 - Learning emergent design, evolutionary design

Architect to Craftsperson

How will I know the learning levels in software craftsmanship?

Dreyfus Model of Skills Acquisition

- Novice** - Needs to be told exactly what to do. No context to work from.
- Advanced Beginner** - Has more context, but needs rigid guidelines
- Competent** - Questions reasoning behind the tasks and can see consequences
- Proficient** - Still relies on rules, but can separate what's important
- Expert** - Works mainly on intuition, except when problems occur

Finding Your Own Identity is about **Metamorphosis** (Shu – Ha – Ri)

From Architect (Crawling Caterpillar) to Leader Craftsperson (Soaring Butterfly)

Creative Cycle

To Follow Agile → To Be agile

Responsibility

Apprenticeship



Pride
("Signing"
Your Work)

Collaboration

**Continuous
Learner**

Argumentation

Novice

Advanced
Beginner

Competent

Proficient

**Deliberate
Practice**

Conflict Mining

Expert

Team Intelligence

Right Attitude

**Psychological
Distance
Solvent**

Follower → Volunteer → Mentor
守破離

**Community
Contributor**

Agility in Enterprise Architecture

A Healthcare Systems Case Study

The Burning Issue Too Much Information!

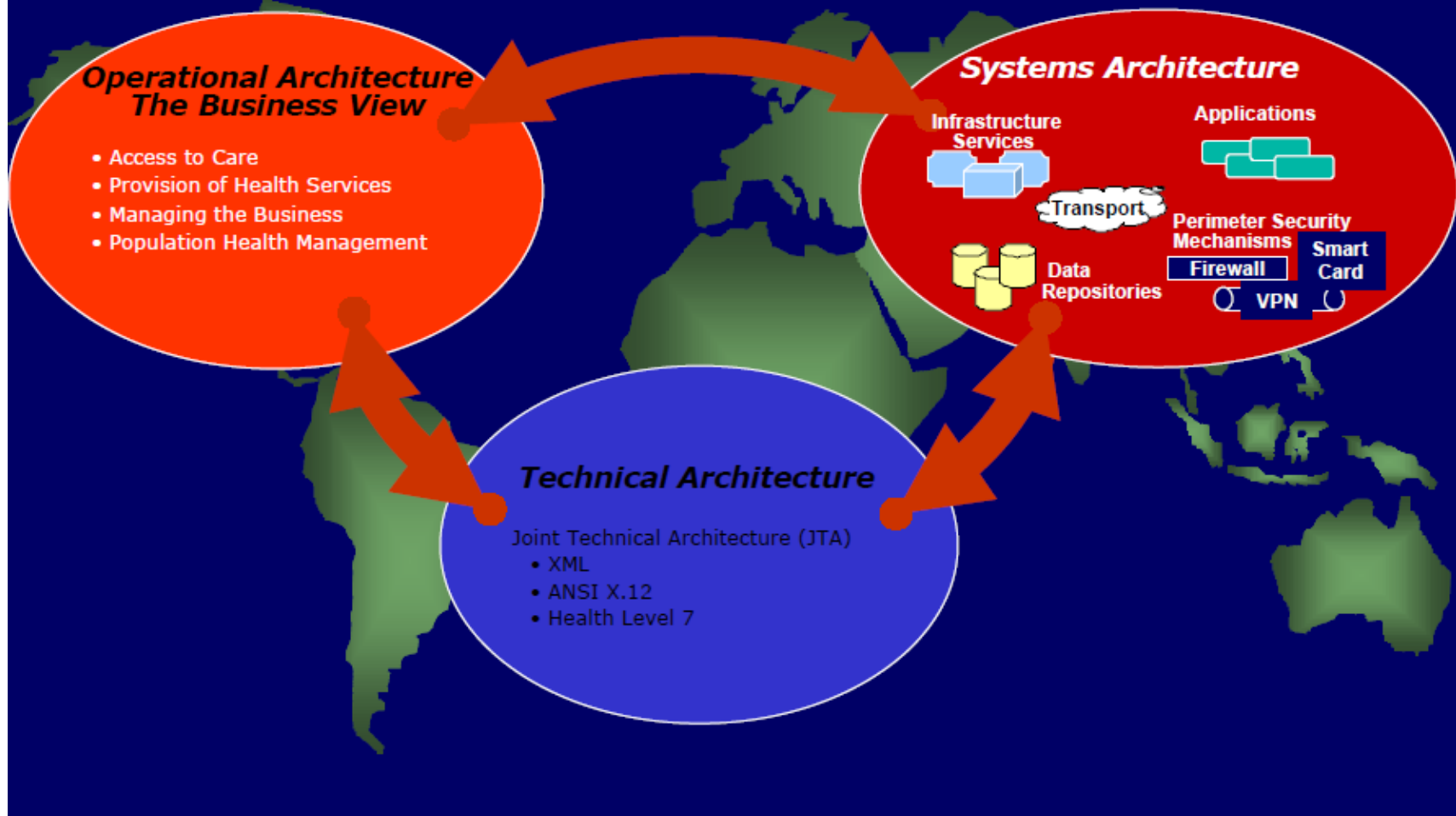
“A weekday edition of the *New York Times* contains more information than the average person was likely to come across in a lifetime in seventeenth century England.”

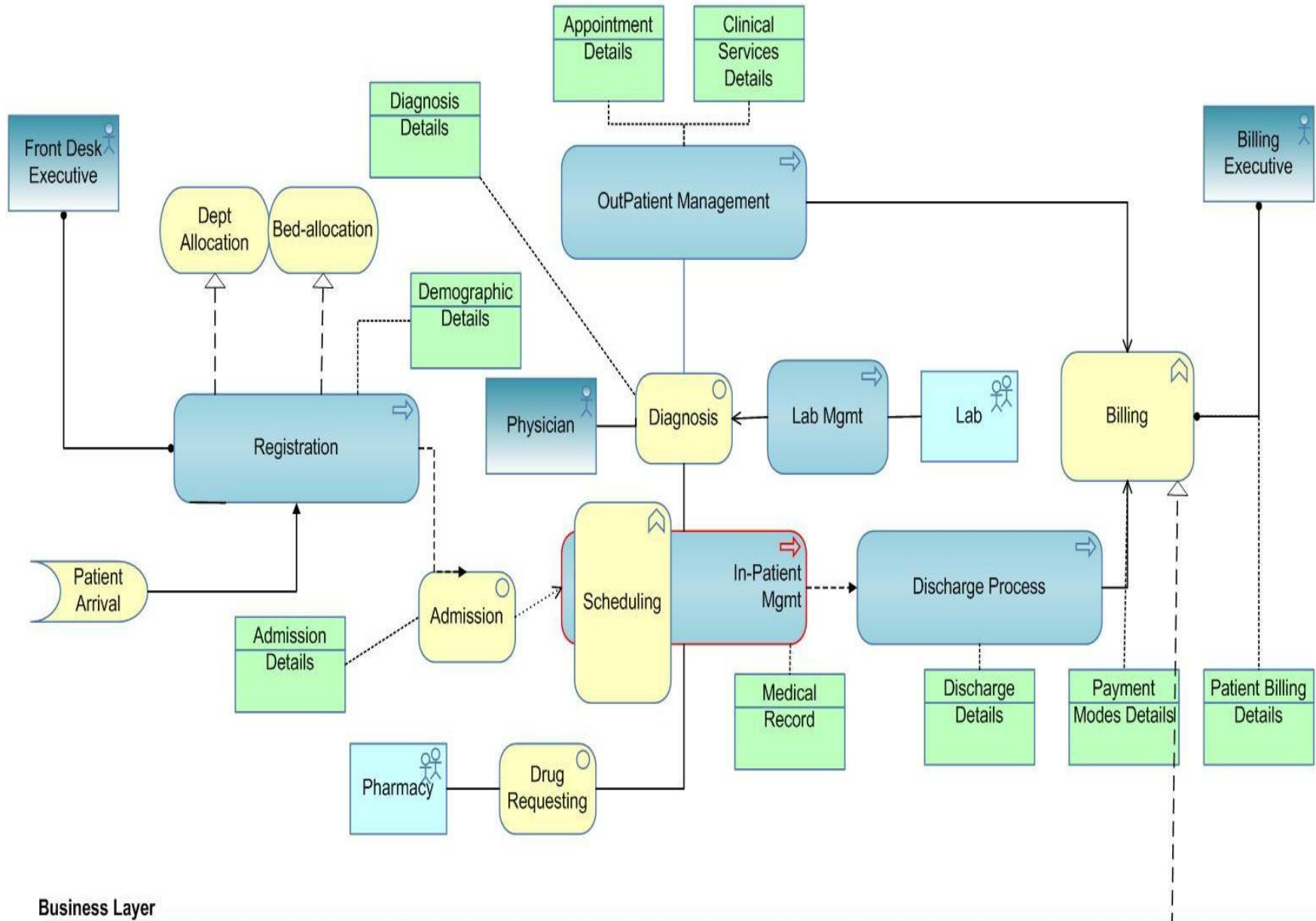
The complexity of managing a large, geographically dispersed healthcare is beyond the capacity of the unassisted human brain

Benefits of an Enterprise Architecture

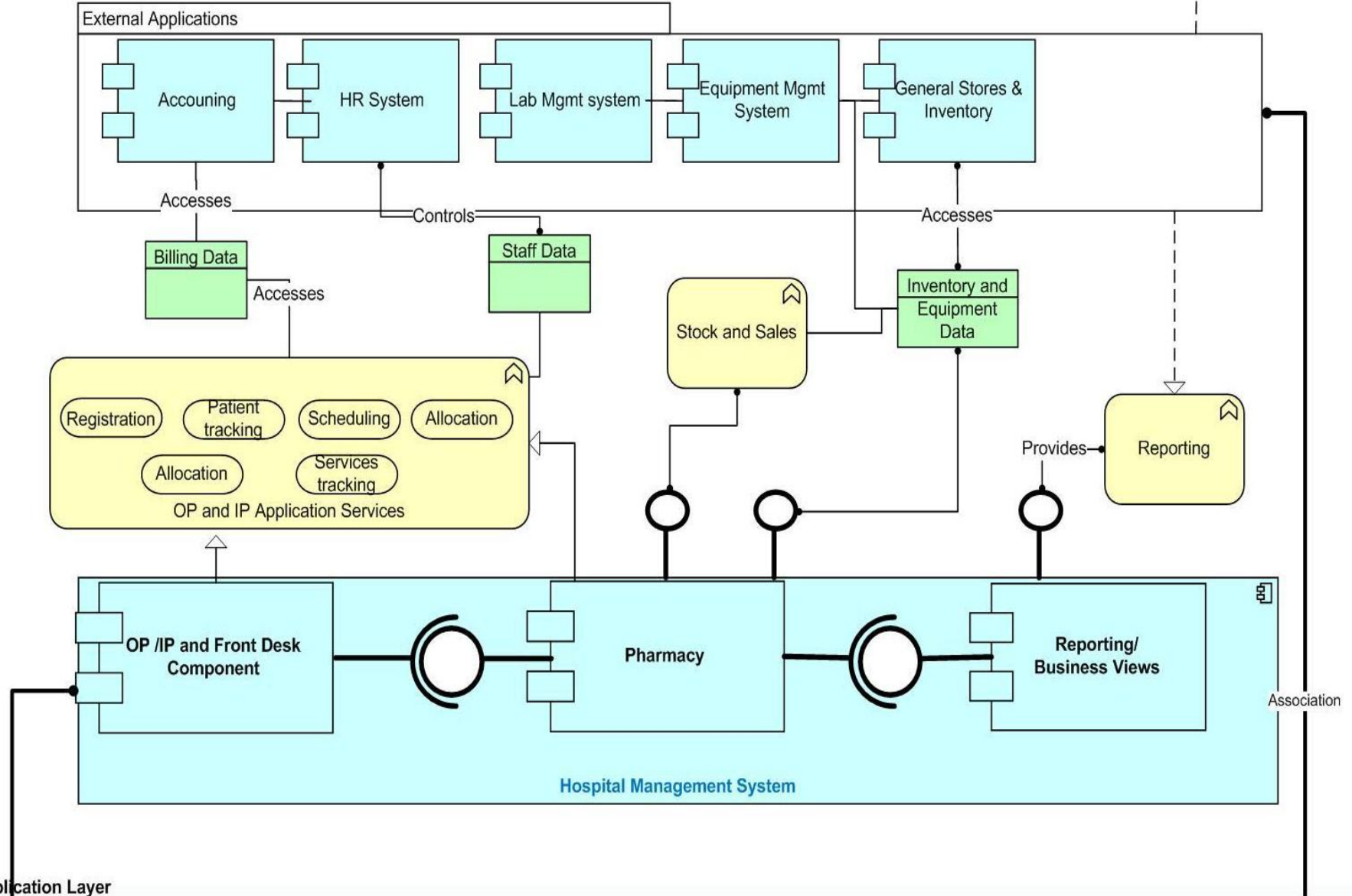
- EA Business (Operational) Component: Critical
 - ✓ Documents your organizational goals and business processes and aligns resources with them
 - ✓ Serves as a framework for knowledge management
- EA IM/IT Components (System and Technical): Supportive
 - ✓ Aligns IT systems with organizational goals
 - ✓ Crucial to interoperability of systems
 - ✓ Avoids duplication of functions in different systems
 - ✓ Allows for development of Common Services

Enterprise Architecture Strategy

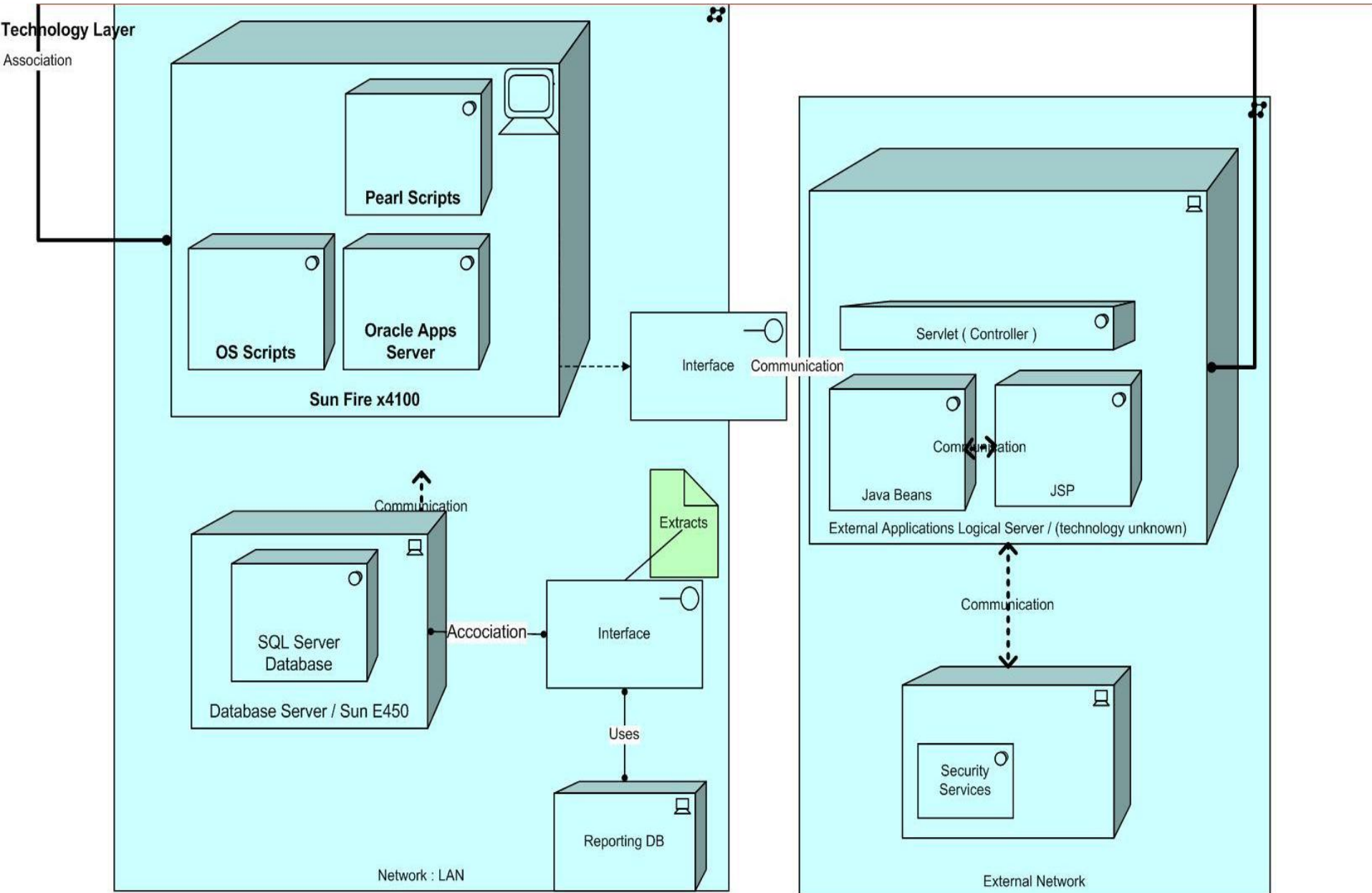




Application Layer



Application Layer





Thank You

Ajay Danait

Ajay.Danait@stixis.com

www.stixis.com