UNIT-V
PART - B

Project Organizations and Responsibilities: Line-of-Business Organizations, Project Organizations, Evolution of Organizations.

Line-of-Business Organizations

- Below figure maps roles and responsibilities to a default line-of-business organization. This structure can be tailored to specific circumstances.

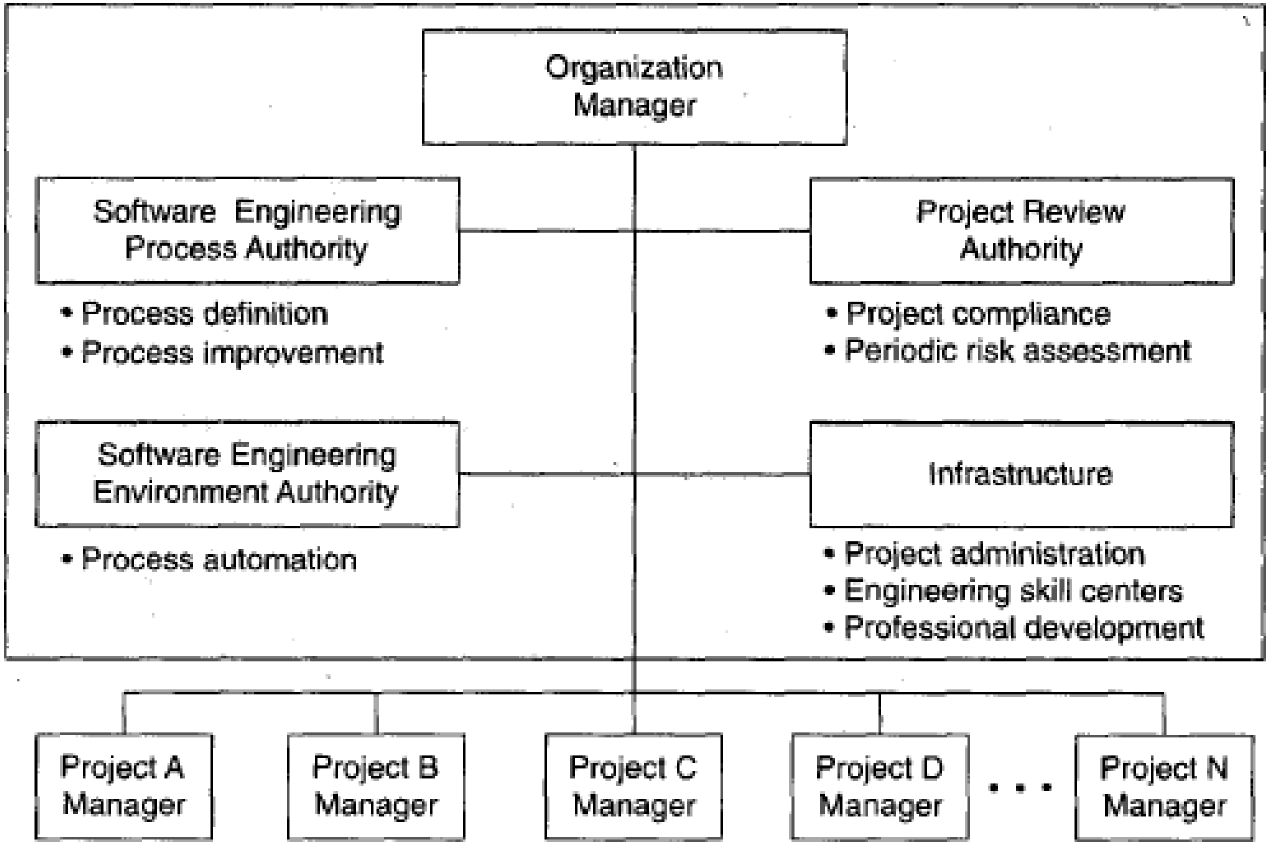


Fig: Default roles in a software line-of-business organization

Line-of-Business Organizations

- The main features of default organization are as follows:
  - Responsibility for process definition and maintenance is specific to a cohesive line of business, where process commonality makes sense.
  - Responsibility for process automation is an organizational role and is equal in importance to the process definition role.
  - Organizational roles may be fulfilled by a single individual or several different teams, depending on the scale of the organization.

Software Engineering Process Authority

- The **Software Engineering Process Authority (SEPA)** facilitates the exchange of information and process guidance both to and from project practitioners.
- This role is accountable to the organization general manager for maintaining a current assessment of the organization’s process maturity and its plan for future improvements.
- The SEPA must help initiate and periodically assess project processes.
- It takes on responsibility and accountability for the process definition and its maintenance (modification, improvement, technology insertion).

Project Review Authority

- The **Project Review Authority (PRA)** is the single individual responsible for ensuring that a software project complies with all organizational and business unit software policies, practices, and standards.
- A software project manager is responsible for meeting the requirements of a contract or some other project compliance standard, and is also accountable to the PRA.
- The PRA reviews both the projects conformance to contractual obligations and the project organizational policy obligations.

Software Engineering Environment Authority

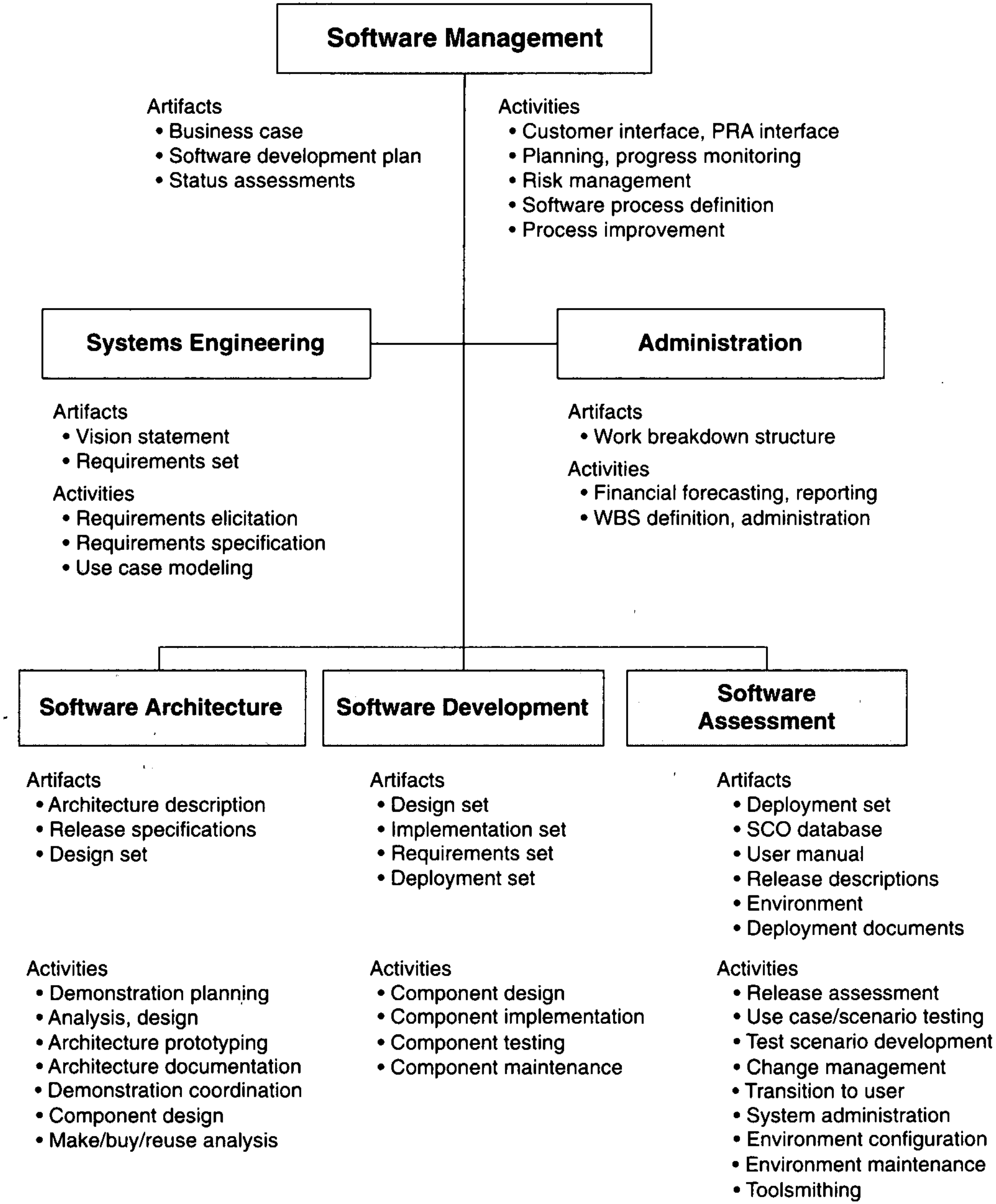
- The **Software Engineering Environment Authority (SEEA)** is responsible for automating the organization’s process, maintaining the organization’s standard environment, training projects to use the environment and maintaining organization-wide reusable assets.
- The SEEA role is necessary to achieve a significant return on investment for a common process.

Infrastructure

- An **organization's infrastructure** provides human resources support, project-independent research and development, and other capital software engineering assets.
- The typical components of the organizational infrastructure are as follows:
- **Project administration**: time accounting system; contracts, pricing, terms and conditions; corporate information systems integration
- **Engineering skill centers**: custom tools repository and maintenance, bid and proposal support, independent research and development
- **Professional development**: internal training boot camp, personnel recruiting, personnel skills database maintenance, literature and assets library, technical publications

Project Organizations

- Below figure shows a default project organization and maps project-level roles and responsibilities.
- This structure can be tailored to the size and circumstances of the specific project organization.



Software Management

Artifacts
- Business case
- Software development plan
- Status assessments

Activities
- Customer interface, PRA interface
- Planning, progress monitoring
- Risk management
- Software process definition
- Process improvement

Systems Engineering

Artifacts
- Vision statement
- Requirements set

Activities
- Requirements elicitation
- Requirements specification
- Use case modeling

Administration

Artifacts
- Work breakdown structure

Activities
- Financial forecasting, reporting
- WBS definition, administration

Software Architecture

Artifacts
- Architecture description
- Release specifications
- Design set

Activities
- Demonstration planning
- Analysis, design
- Architecture prototyping
- Architecture documentation
- Demonstration coordination
- Component design
- Make/buy/reuse analysis

Software Development

Artifacts
- Design set
- Implementation set
- Requirements set
- Deployment set

Activities
- Component design
- Component implementation
- Component testing
- Component maintenance

Software Assessment

Artifacts
- Deployment set
- SCO database
- User manual
- Release descriptions
- Environment
- Deployment documents

Activities
- Release assessment
- Use case/scenario testing
- Test scenario development
- Change management
- Transition to user
- System administration
- Environment configuration
- Environment maintenance
- Toolsmithing

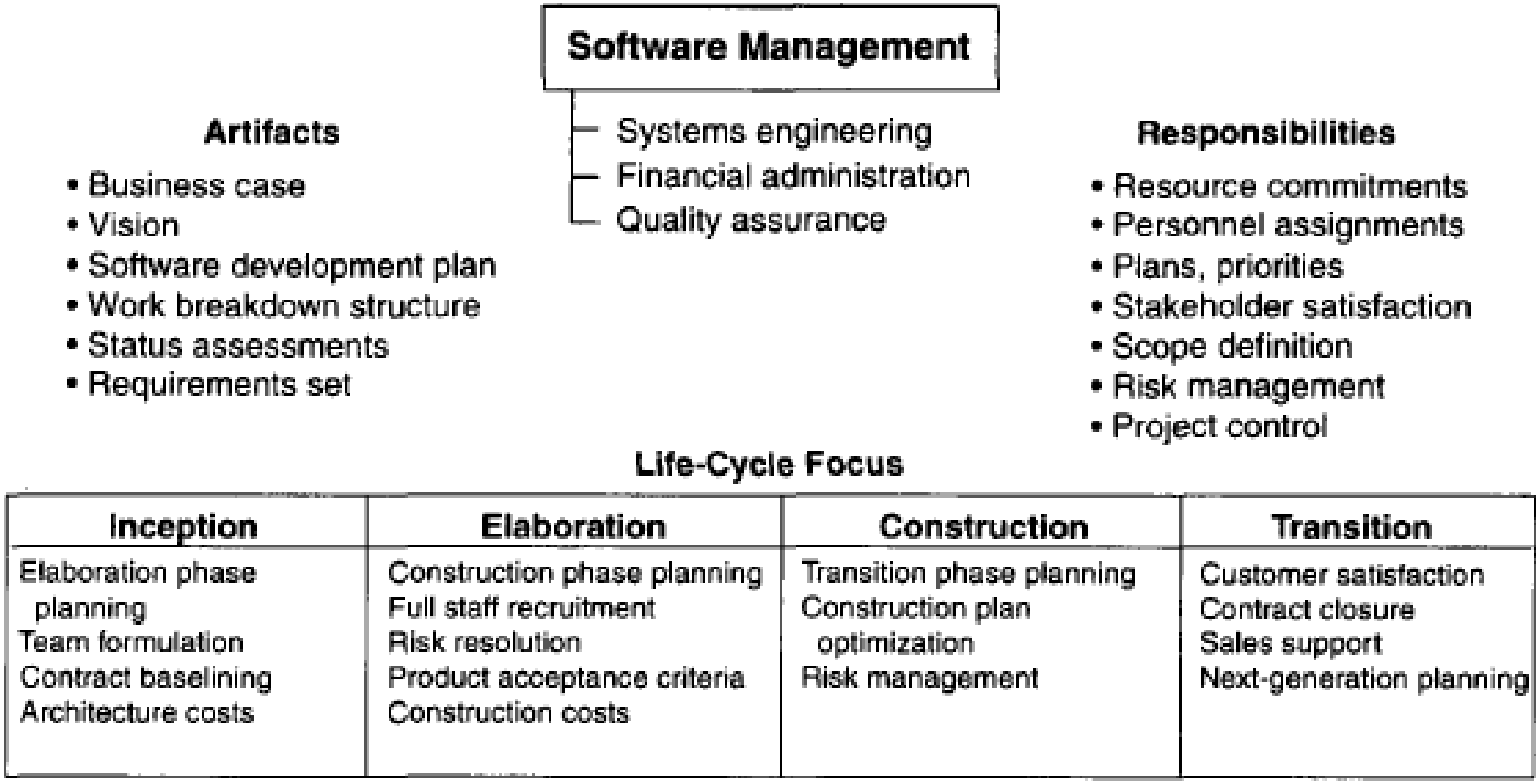
Project Organizations

- The main features of the default organization are as follows:
  - The **project management team** is an active participant, responsible for producing as well as managing.
  - The **architecture team** is responsible for real artifacts and for the integration of components, not just for staff functions.
  - The **development team** owns the component construction and maintenance activities.
  - The **assessment team** is separate from development team. This structure fosters an independent quality perspective and focuses a team on testing and product evaluation activities concurrent with on-going development.
  - The **Quality** is everyone’s job, integrated into all activities and checkpoints. Each team takes responsibility for a different quality perspective.

Software Management Team

- Most projects are overconstrained.
- Schedules, costs, functionality, and quality expectations are highly interrelated and require continuous negotiation among multiple stakeholders who have different goals.
- The software management team carries the burden of delivering win conditions to all stakeholders.
- In this regard, the software project manager spends every day worrying about ***balance***.
- Below figure shows the focus of software management team activities over the project life cycle.

Software Management Team



Software Management

- Systems engineering
- Financial administration
- Quality assurance

Artifacts

- Business case
- Vision
- Software development plan
- Work breakdown structure
- Status assessments
- Requirements set

Responsibilities

- Resource commitments
- Personnel assignments
- Plans, priorities
- Stakeholder satisfaction
- Scope definition
- Risk management
- Project control

Life-Cycle Focus

Inception	Elaboration	Construction	Transition
Elaboration phase planning Team formulation Contract baselining Architecture costs	Construction phase planning Full staff recruitment Risk resolution Product acceptance criteria Construction costs	Transition phase planning Construction plan optimization Risk management	Customer satisfaction Contract closure Sales support Next-generation planning

Figure: Software management team activities

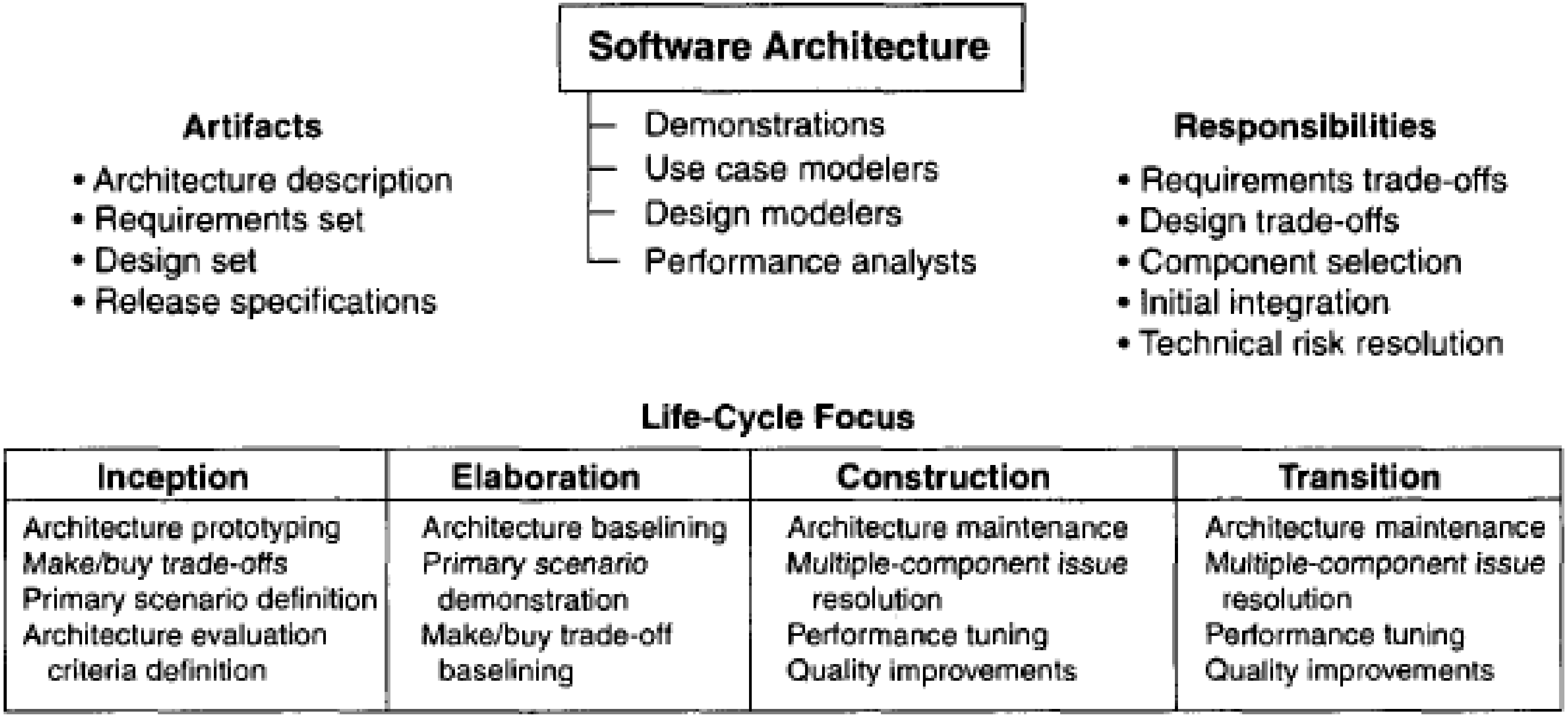
Software Management Team

- The software management team is responsible for planning the effort, conducting the plan, and adapting the plan to changes in the understanding of the requirements or the design.
- The team takes ownership of resource management and project scope, and sets operational priorities across the project life cycle.
- The software management team takes ownership of all aspects of quality.

Software Architecture Team

- The software architecture team is responsible for the architecture.
- This responsibility encompasses the engineering necessary to specify a complete bill of materials for the software and the engineering necessary to make significant make/buy trade-offs so that all custom components are elaborated to the extent that construction/assembly costs are highly predictable.
- Below figure shows the focus of software architecture team activities over the project life cycle.

Software Architecture Team



Software Architecture

- Demonstrations
- Use case modelers
- Design modelers
- Performance analysts

Artifacts

- Architecture description
- Requirements set
- Design set
- Release specifications

Responsibilities

- Requirements trade-offs
- Design trade-offs
- Component selection
- Initial integration
- Technical risk resolution

Life-Cycle Focus

Inception	Elaboration	Construction	Transition
Architecture prototyping Make/buy trade-offs Primary scenario definition Architecture evaluation criteria definition	Architecture baselining Primary scenario demonstration Make/buy trade-off baselining	Architecture maintenance Multiple-component issue resolution Performance tuning Quality improvements	Architecture maintenance Multiple-component issue resolution Performance tuning Quality improvements

Figure: Software architecture team activities

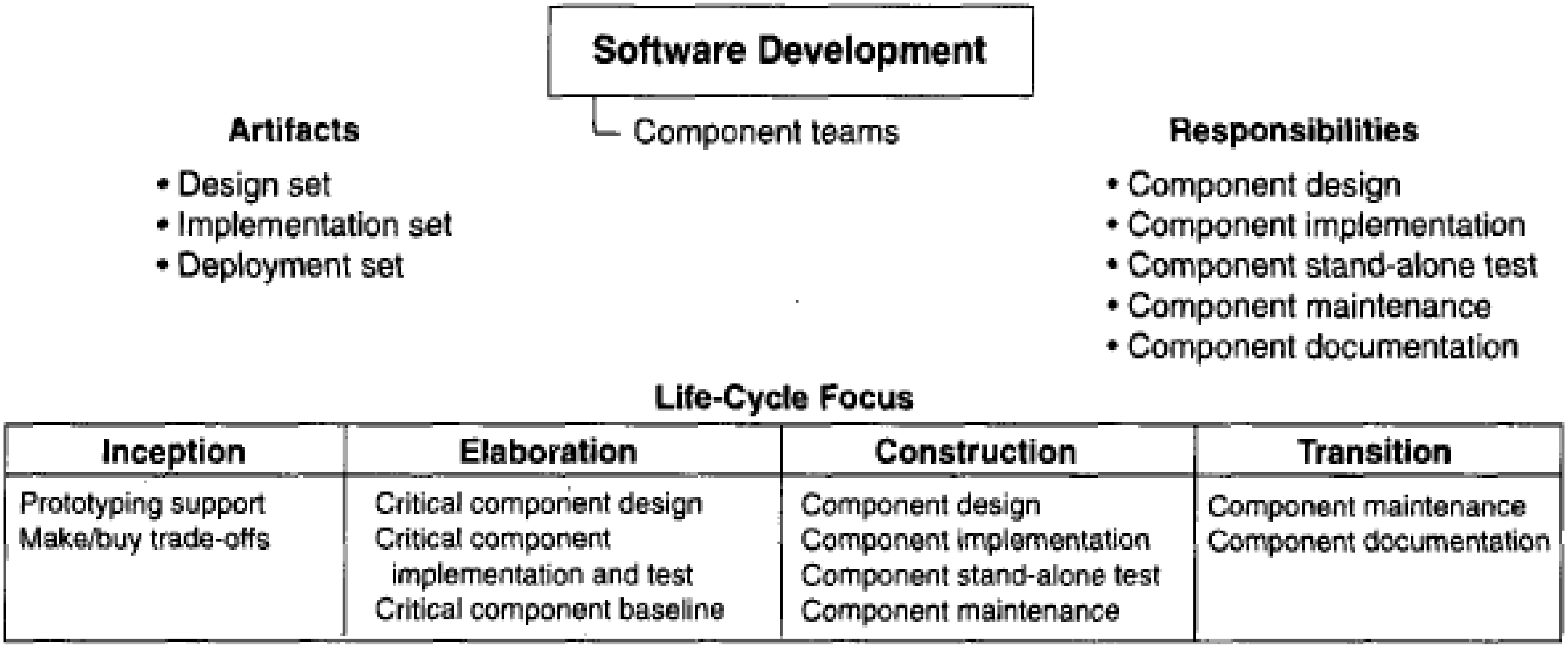
Software Architecture Team

- To succeed, the architecture team must include a fairly broad level of expertise, including the following:
  - Domain experience to produce an acceptable design view (architecturally significant elements of the design model) and use case view (architecturally significant elements of the use case model)
  - Software technology experience to produce an acceptable process view (concurrency and control thread relationships among the design, component, and deployment models), component view (structure of the implementation set), and deployment view (structure of the deployment set)

Software Development Team

- Below figure shows the focus of software development team activities over the project life cycle.
- The software development team is the most application-specific group. In general, the software development team comprises several subteams dedicated to groups of components that require a common skill set.
- Typical skill sets include the following:
  - **Commercial component**: specialists with detailed knowledge of commercial components central to a system's architecture
  - **Database**: specialists with experience in the organization, storage, and retrieval of data

Software Development Team



Software Development

Component teams

Artifacts

- Design set
- Implementation set
- Deployment set

Responsibilities

- Component design
- Component implementation
- Component stand-alone test
- Component maintenance
- Component documentation

Life-Cycle Focus

Inception	Elaboration	Construction	Transition
Prototyping support Make/buy trade-offs	Critical component design Critical component implementation and test Critical component baseline	Component design Component implementation Component stand-alone test Component maintenance	Component maintenance Component documentation

Figure: Software development team activities

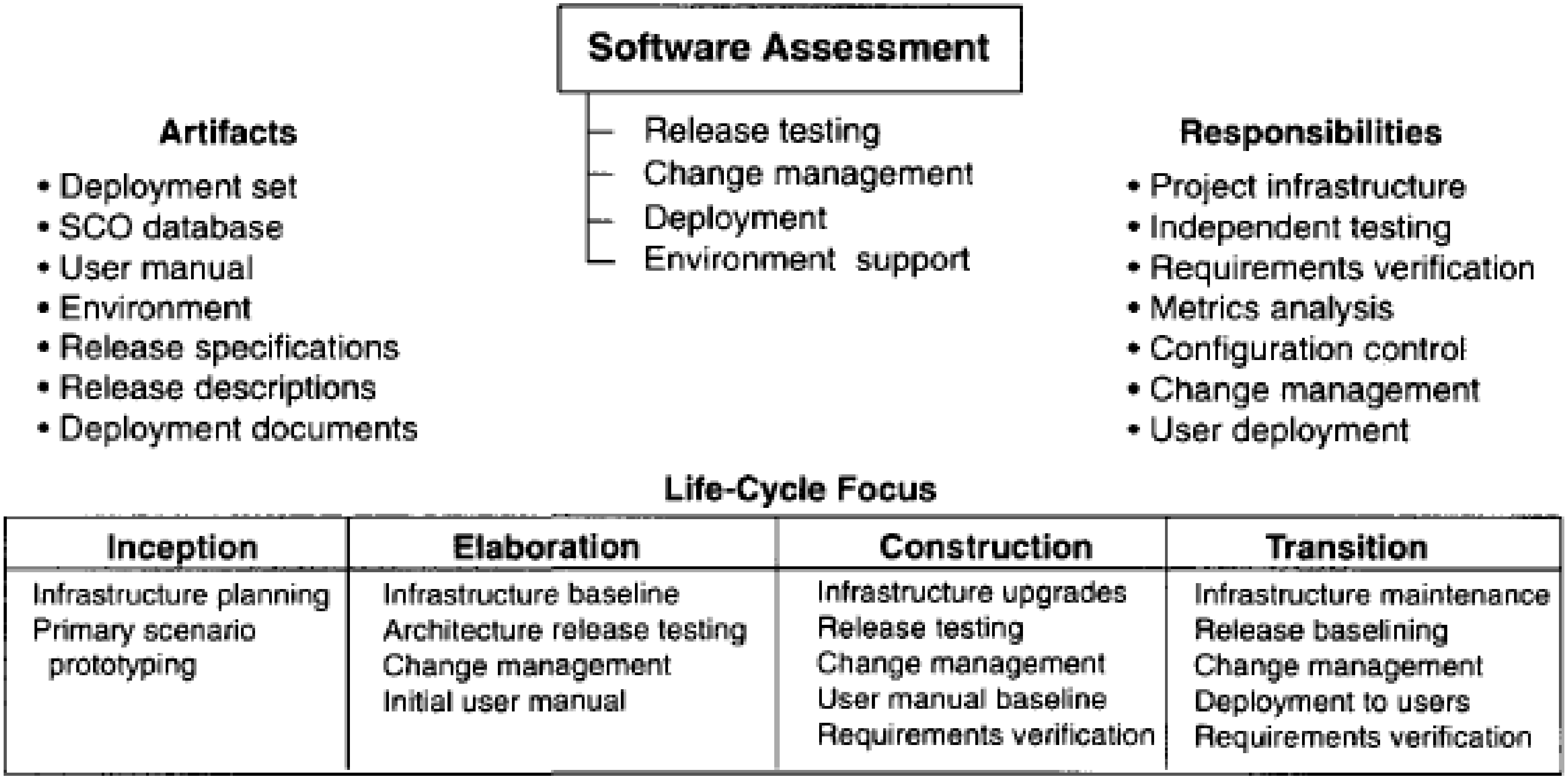
Software Development Team

- **Graphical user interfaces**: specialists with experience in the display organization, data presentation, and user interaction necessary to support human input, output, and control needs
- **Operating systems and networking**: specialists with experience in the execution of multiple software objects on a network of hardware resources, including all the typical control issues associated with initialization, synchronization, resource sharing, name space management, reconfiguration, termination, and interobject communications
- **Domain applications**: specialists with experience in the algorithms, application processing, or business rules specific to the system

Software Assessment Team

- Below figure shows the focus of software assessment team activities over the project life cycle.
- There are two reasons for using an independent team for software assessment.
- The first-has to do with ensuring an independent quality perspective.
- A more important reason for using an independent test team is to exploit the concurrency of activities.

Software Assessment Team



Software Assessment

- Release testing
- Change management
- Deployment
- Environment support

Artifacts

- Deployment set
- SCO database
- User manual
- Environment
- Release specifications
- Release descriptions
- Deployment documents

Responsibilities

- Project infrastructure
- Independent testing
- Requirements verification
- Metrics analysis
- Configuration control
- Change management
- User deployment

Life-Cycle Focus

Inception	Elaboration	Construction	Transition
Infrastructure planning Primary scenario prototyping	Infrastructure baseline Architecture release testing Change management Initial user manual	Infrastructure upgrades Release testing Change management User manual baseline Requirements verification	Infrastructure maintenance Release baselining Change management Deployment to users Requirements verification

Figure: Software development team activities

Software Assessment Team

- A modern process should employ use-case-oriented or capability-based testing (which may span many components) organized as a sequence of builds and mechanized via two artifacts:

1. Release specification (the plan and evaluation criteria for a release)
2. Release description (the results of a release)

- Each release may encompass several (perhaps incomplete) components, because integration is proceeding continuously.
- Evaluation criteria will document what the customer may expect to see at a major milestone, and release descriptions will substantiate the test results.

Evolution of Organizations

- The project organization represents the architecture of the team and needs to evolve consistent with the project plan captured in the work breakdown structure.
- Below figure illustrates how the team's center of gravity shifts over the life cycle, with about 50% of the staff assigned to one set of activities in each phase.

Evolution of Organizations

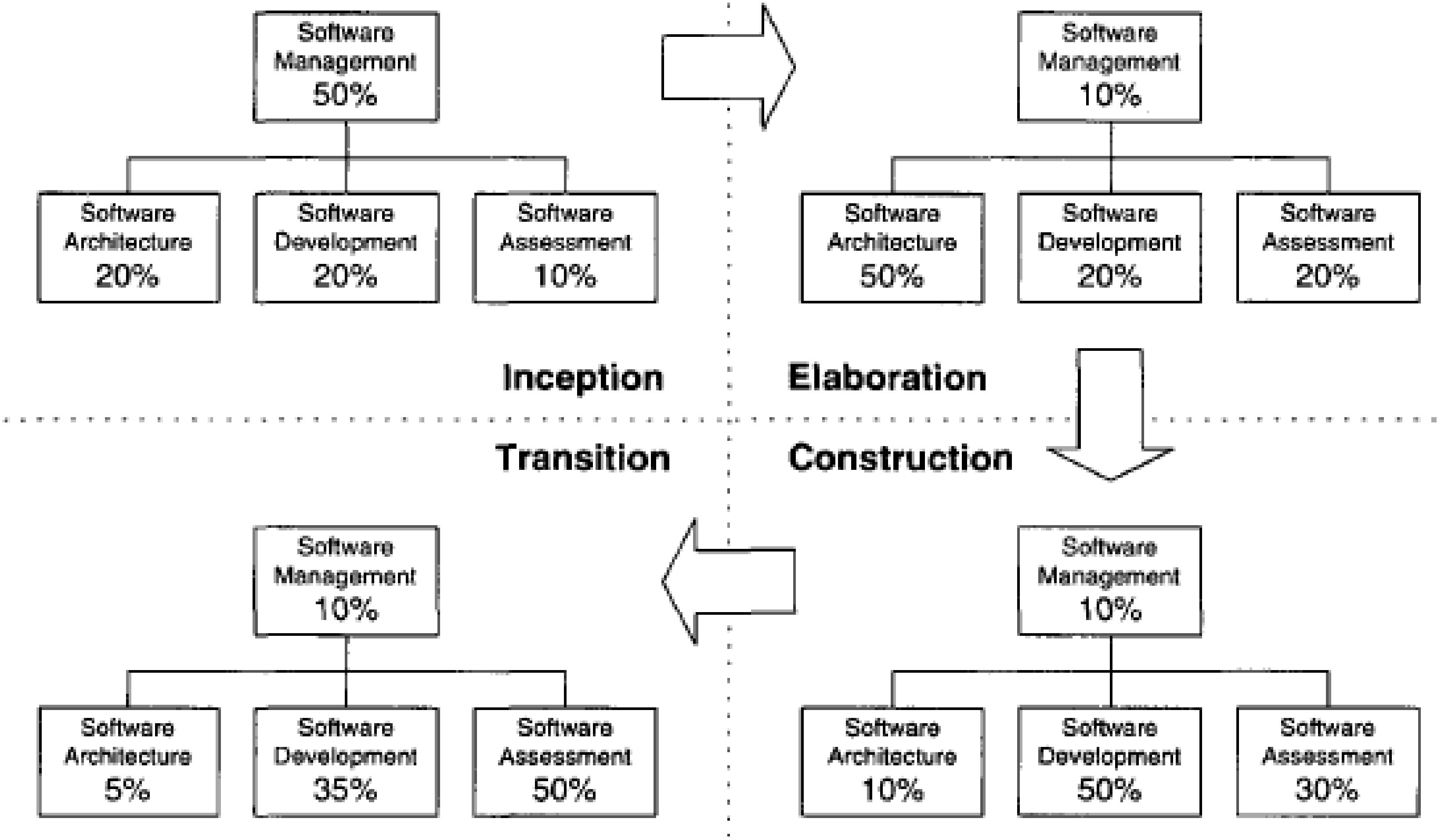


Figure: Software project team evolution over the life cycle

Evolution of Organizations

- A different set of activities is emphasized in each phase, as follows:
- **Inception team**: an organization focused on planning, with enough support from the other teams to ensure that the plans represent a consensus of all perspectives

Evolution of Organizations

- **Elaboration team**: an architecture-focused organization in which the driving forces of the project reside in the software architecture team and are supported by the software development and software assessment teams as necessary to achieve a stable architecture baseline

Evolution of Organizations

- **Construction team**: a fairly balanced organization in which most of the activity resides in the software development and software assessment teams
- **Transition team**: a customer-focused organization in which usage feedback drives the deployment activities