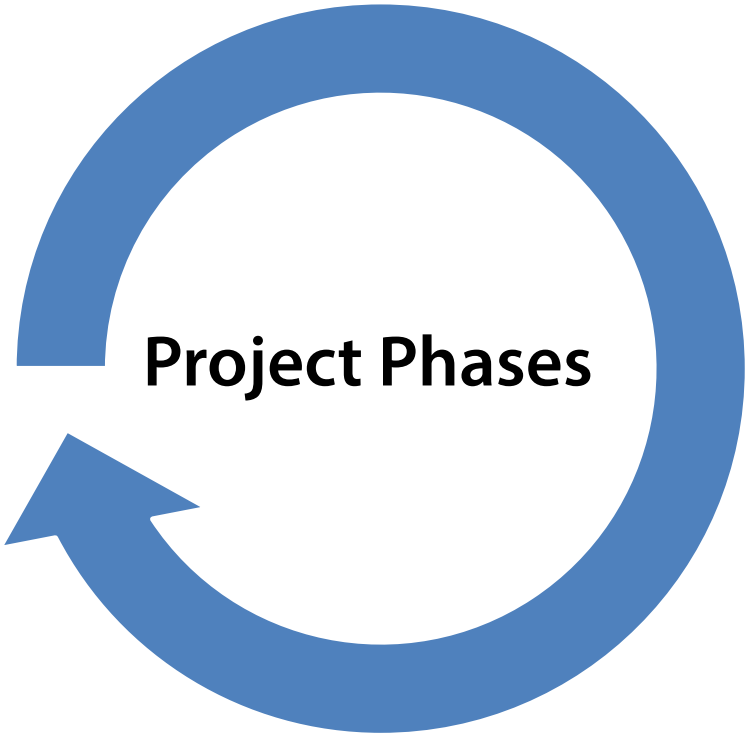
The Software Architect's Role in the project life cycle

Chris Simmons
www.avidsoftware.com
chris.simmons@avidsoftware.com

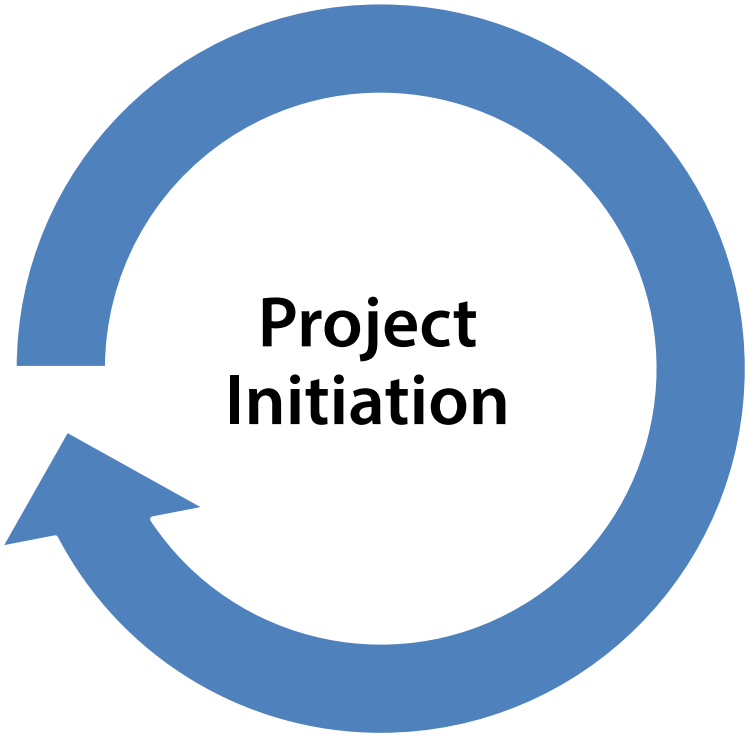
pluralsight
hardcore developer training

The architect's role in the project life cycle



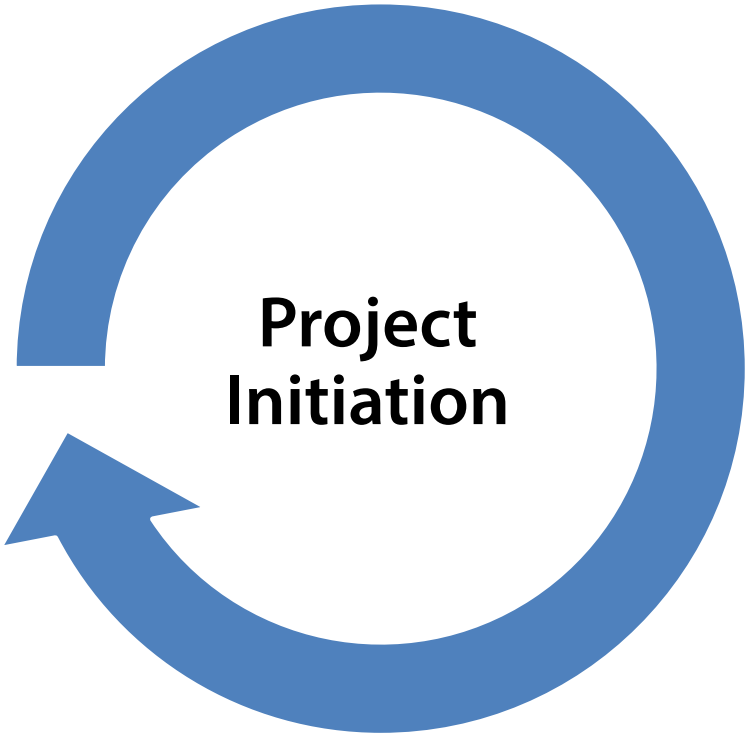
- Project Initiation
- Assembling the team
- Requirements
- Design
- Construction
- Testing
- Implementation

The architect's role in the project life cycle



- **Projects are initiated by LOB**
  - New product
  - Enhancement to existing product
- **Projects assigned to architect most experienced in:**
  - Technology
  - Business domain
- **Multiple architects may be assigned**
- **Outline a viable solution**
- **Get comfortable with ambiguity**
- **Listen to your stakeholders**
  - Understand business goals

The architect's role in the project life cycle



- **Identify the most complex pieces of the solution**
  - Interop points of solution
  - Most rigid & time consuming
- **Identify portions of the solution that will most greatly impact:**
  - Timeline
  - Cost
- **Architecturally significant portions of the project**
- **Consultant providing recommendations**

Project Initiation

- **High level cost and resource estimates**
  - Decide project viability
- **Project Sizing**
  - General Connotation
  - Doesn't imply commitment
  - General expectation to work effort
- **Sometimes project end here**
- **High Level Estimates**
  - Feasibility decision
  - Based on what you know today
  - Revisited later

Project Initiation

- **Project Sizing**
  - Effective Tool
  - Easy to understand
  - Separate into S,M,L buckets
  - Buckets identify resources and duration
- **Examples**
  - Small: <5 people / < 6 months
  - Medium: 5-10 people / 3 to 12 months
  - Large: 25+ / 6 -12+ months
- **Sizing project helps business determine:**
  - Number of resources
  - Project duration
  - Provide enough value

Project Initiation

- Project requirements too vague to provide accurate estimates
- Sizing includes all resources
- Sizing helps business to determine project viability

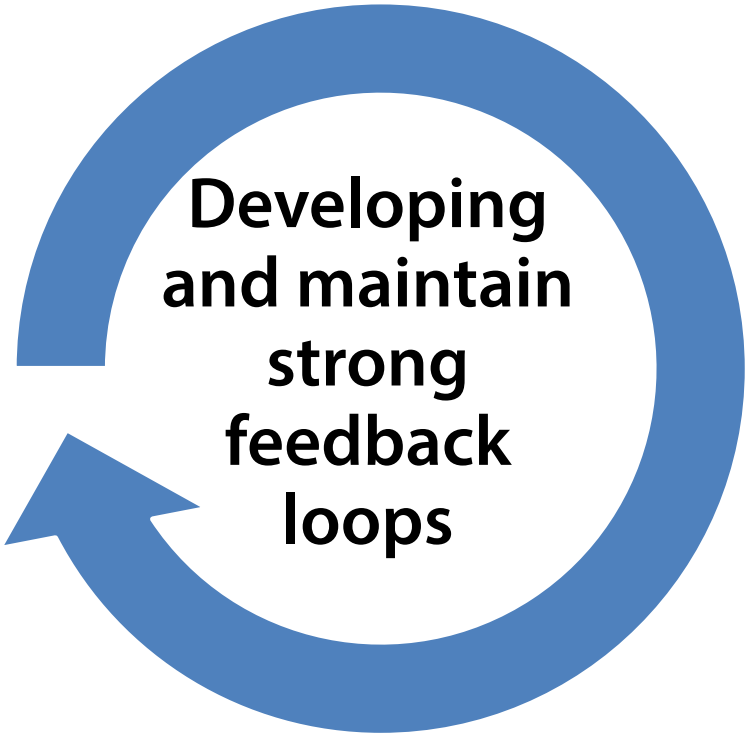
Project Initiation

- **Who are your stakeholders?**
  - organization leaders
  - enterprise architects
  - project managers
  - business users
- **Understand their needs and goals**
  - Identify preferred solutions
  - Business goals
  - Quality attributes
- **Understand the business domain**
  - Interview users
  - Build relationships

Project Initiation

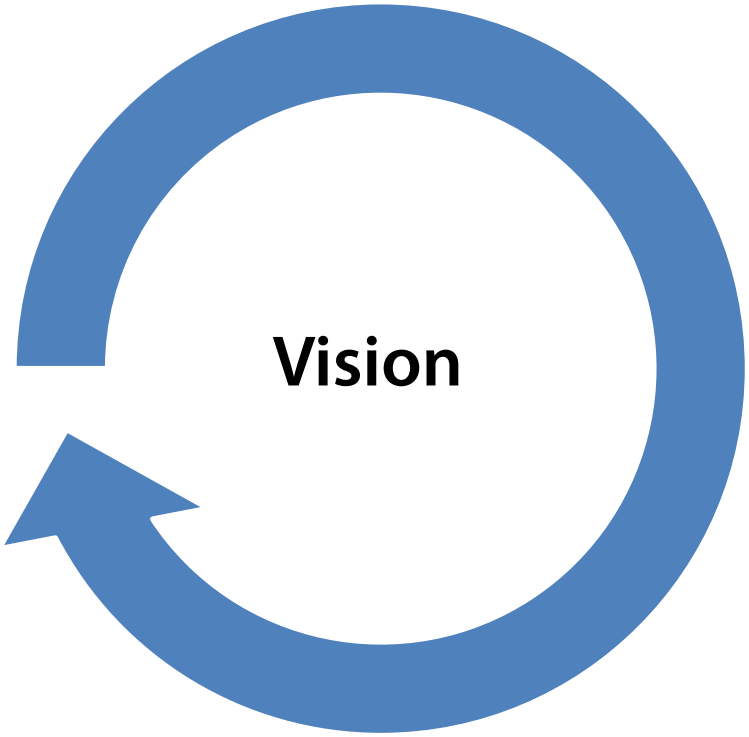
- **Create a general list for your interview**
- **Let the responses drive the interview**
- **Listen intently**
- **Repeat & rephrase questions**
- **Make sure you are clear on answers from all perspectives**
- **Ask why a lot**
  - Challenge the interviewee
  - Helps you understand problem
  - Helps interviewee understand if process adds value
- **Architect must understand problem entirely**
  - Especially angles that seem unimportant

Project Initiation



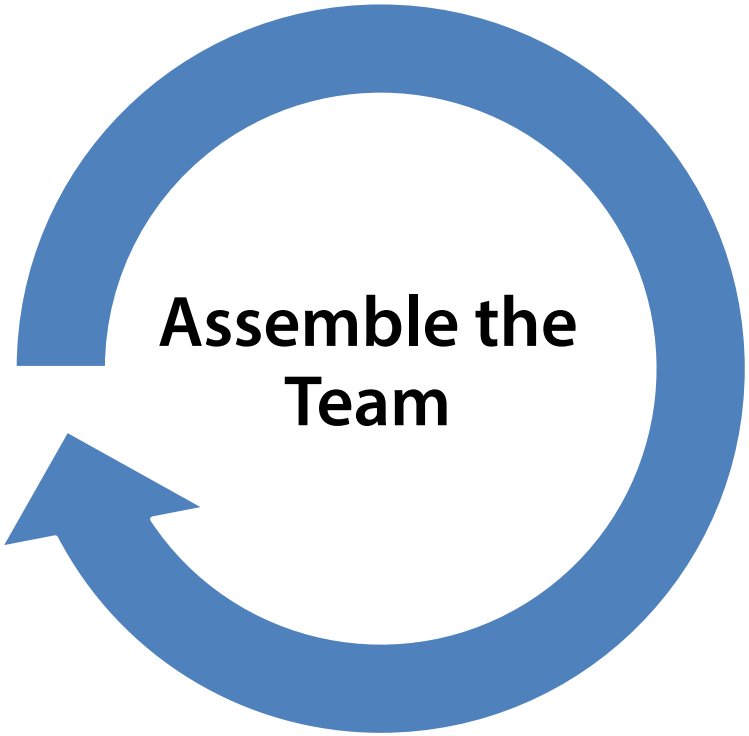
- Make business contacts
- Get to know your stakeholders
- Develop and maintain feedback loops
- These stakeholders are your customers
- They will support the adoption of your solution
- This is the first step in building these important relationships

Project Initiation



- **Technical vision document**
  - Overall vision for the solution
  - Core requirements
  - Key features
  - Quality attributes, constraints, goals
  - Evolves
  - Used as the basis for detailed designs
- **Singular concise vision**
- **Vision drives all technical efforts moving forward**

The architect's role in the project life cycle



- **Assemble your team of experts**
- **Solutions are rarely designed by a single person**
- **Architects rely on large social networks**
- **Comprised of:**
  - Architects
  - Senior Developers
  - Business Domain Experts
- **Lobby heavily for his core team to be assigned to project**
- **Dual Purposes**
  - Create a feedback loop of trusted domain and technology experts

Assemble the Team

- **Collaboration builds:**
  - Acceptance
  - Support
- **Foster a sense of ownership within core team**
- **No place for egos**
- **Architects ability to build a team contributes to projects success**

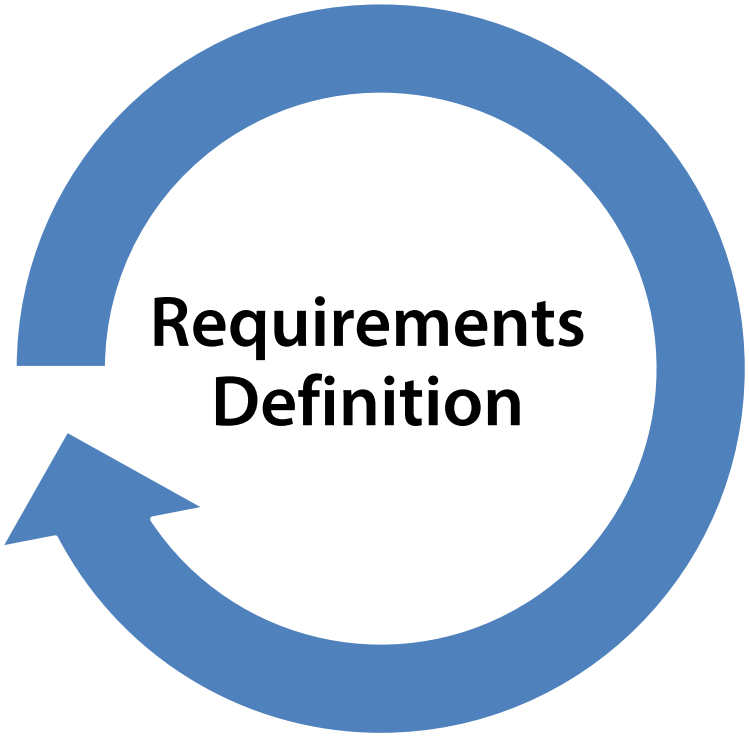
Assemble the Team

- Poll managers for resource availability
- Success is determined by team
- Lobbies for preferred developers
- Longstanding relationships with development managers
- Key to success is the right team
- Architect builds core team
- Architects lobby for best developers
- Lobby both managers and developers

Assemble the Team

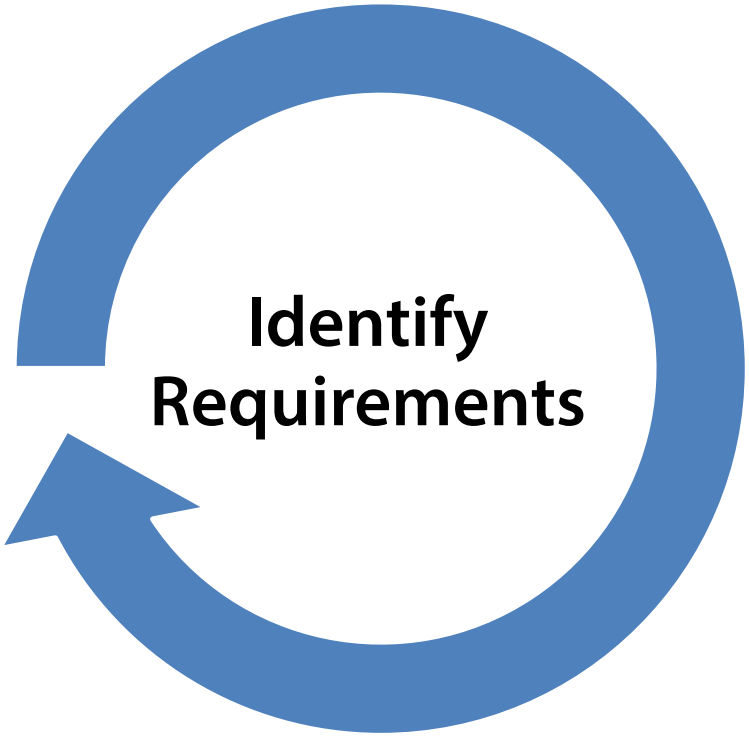
- Team is assembled with collaboration in mind
- Core team is more committed when included in design process
- Collaboration fosters commitment
- Relationships formed here are invaluable

The architect's role in the project life cycle



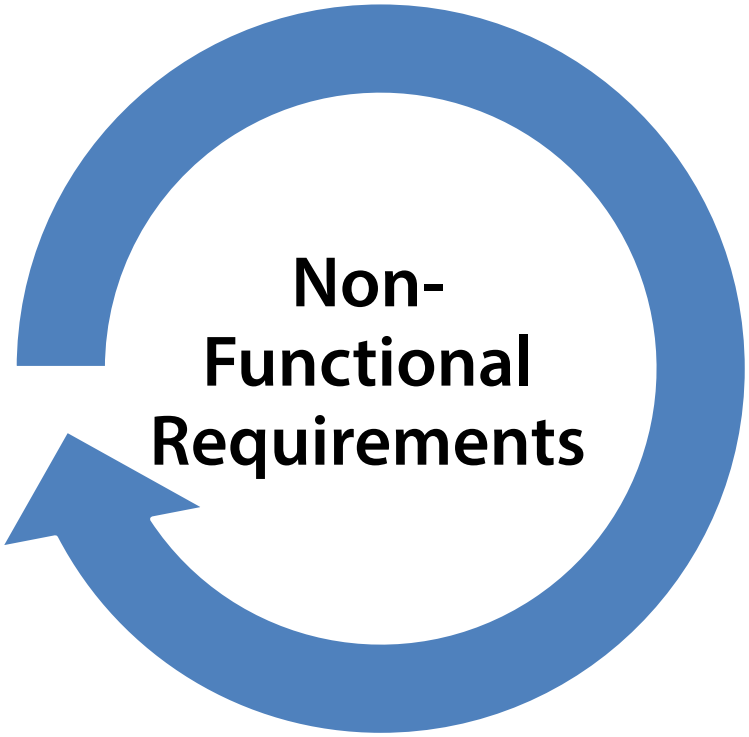
- **Functional requirements**
  - Supportive
- **Non-functional requirements**
  - Active

Requirements Definition



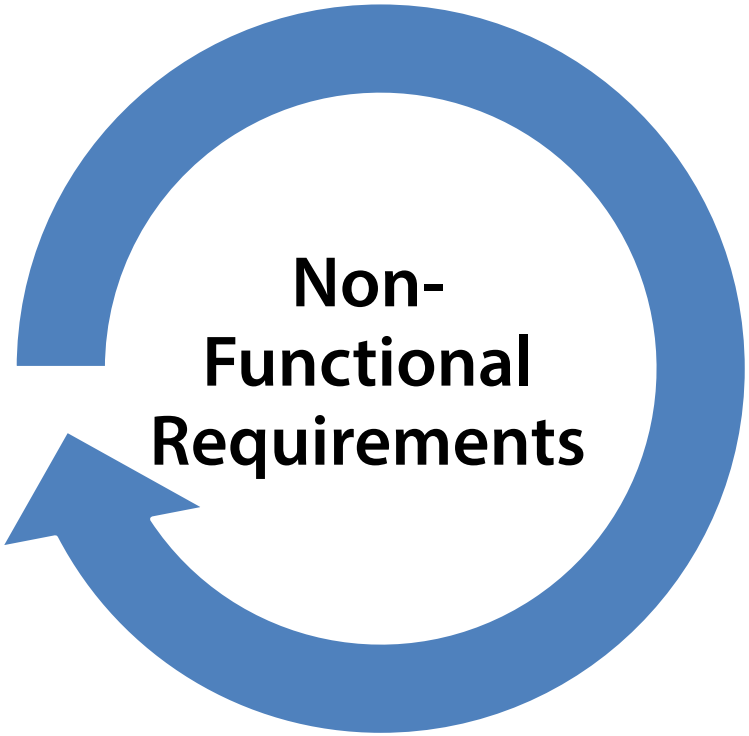
- Analysts don't always capture non-functional requirements
- Functional requirements - What the system *shall do*
- Non-functional requirements - What the system *shall be*

Requirements Definition



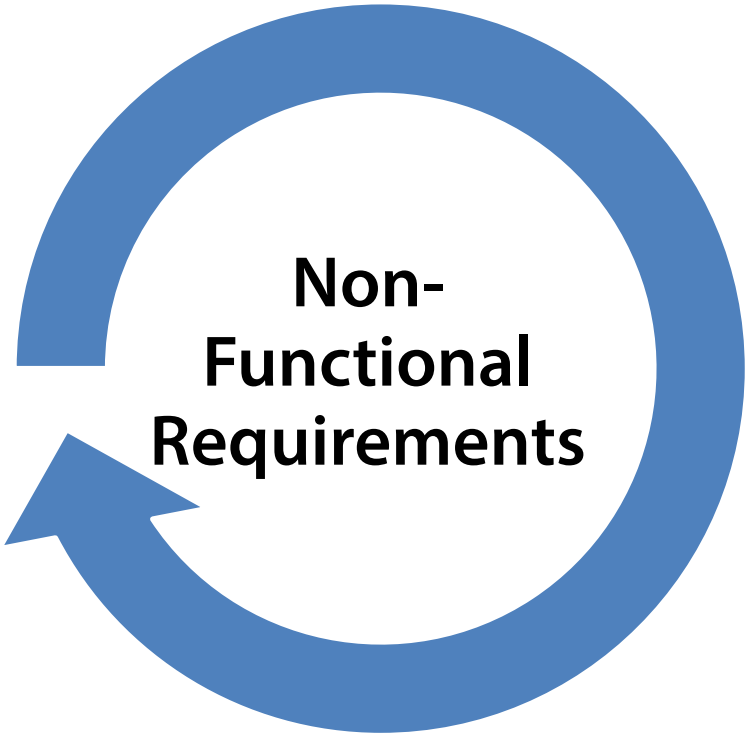
- **Non-functional requirements can greatly impact the architecture**
- **Cross cutting concerns**
- **Quality attributes**
- **Architect actively identifies non-functional requirements**
- **Non-functional requirements guide decisions**
  - Require tough decisions where goals conflict
- **Important to capture non-functional requirements**
  - Explain decisions about architecture

Requirements Definition



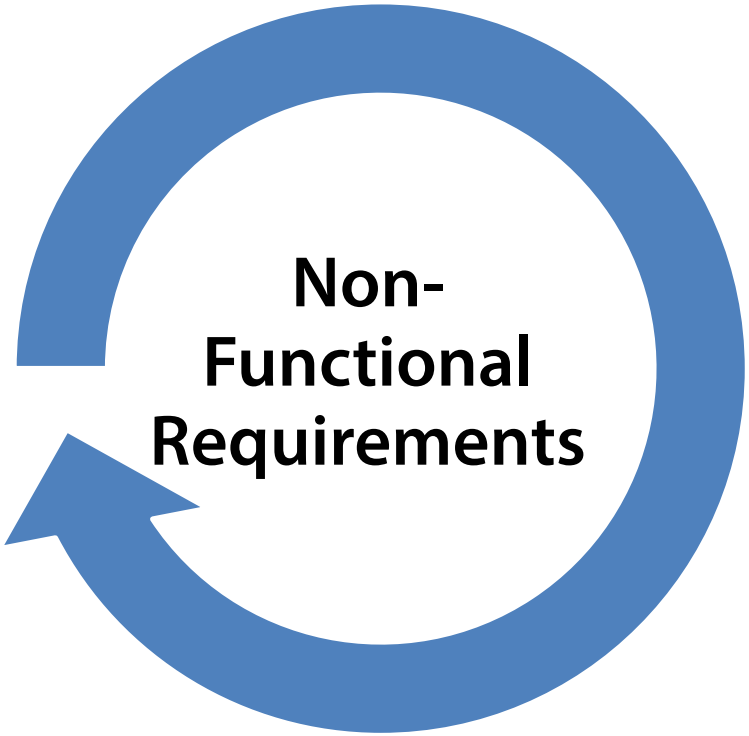
- **Examples are:**
  - Accessibility
  - Availability
  - Configurability
  - Extensibility
  - Performance
  - Maintainability
  - Scalability
  - Security
  - Supportability
  - Testability
  - Usability

Requirements Definition



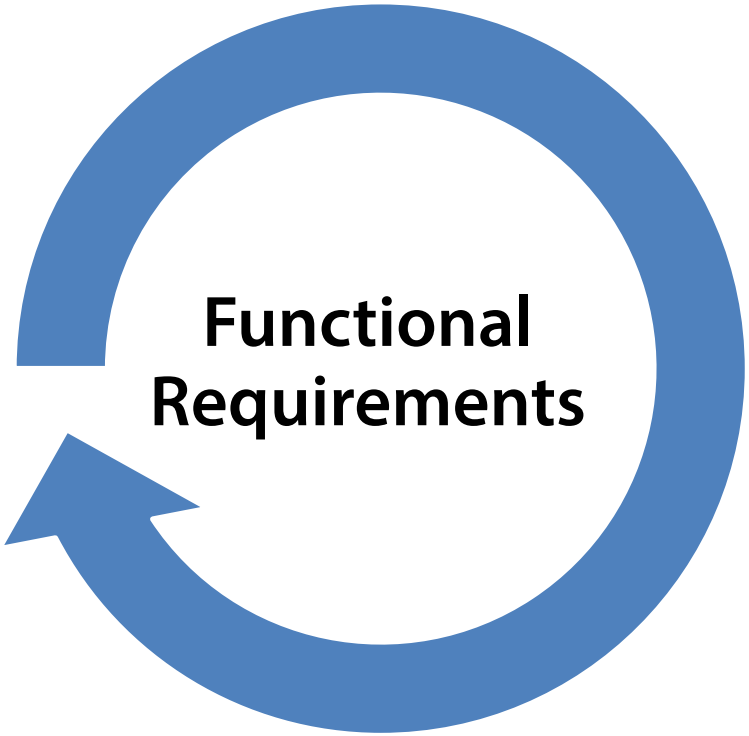
- **Important for architect to understand functional requirements**
  - May be incomplete for design
- **Architect identifies non-functional requirements**
- **Architect interviews stakeholders**
- **Identify as many as possible as early as possible**
  - Helps insure design meets goals
- **Discuss business goals with stakeholders**
  - Deduce non-functional requirements from business goals
- **Ask pointed questions**
  - Use quality attributes list
  - Include cost and time explanations

Requirements Definition



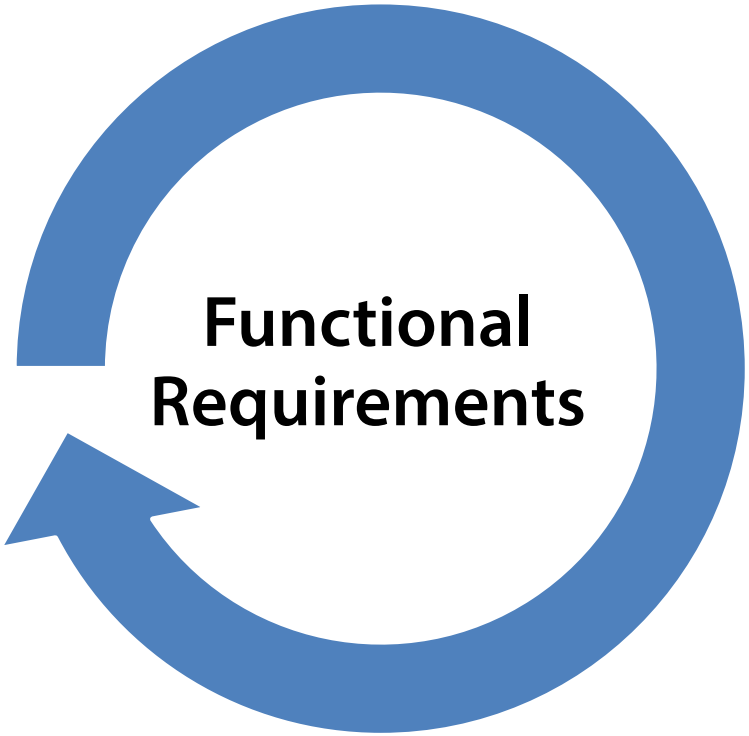
- **Don't just ask if stakeholder wants a quality attribute like scalability**
- **Ask questions like:**
  - How many users are expected to use the system initially?
  - What will this grow to in the future?
  - What will the business impact be if the system is unavailable?
- **Deduce need for scalability from answers to these questions**
  - Weigh against infrastructure and development costs
- **Results is non-functional scalability requirement and why**

Requirements Definition



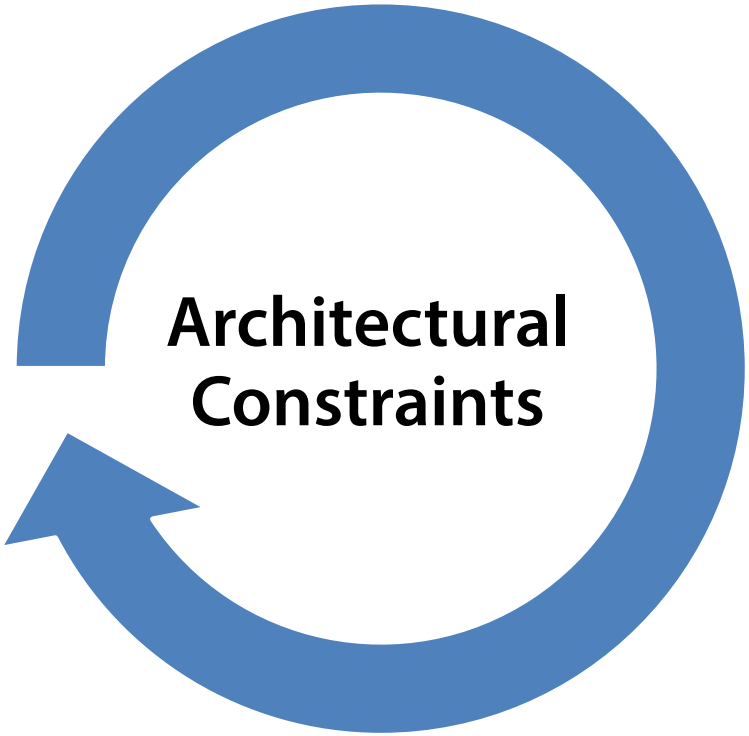
- **Ensure analysts provide good requirements**
- **Basis for architect, development and QA teams**
- **Provide guidance and mentoring to BA's**
- **Good functional requirements define WHAT system should do**
  - Not HOW it should be implemented
- **If HOW is provided by analyst then this is an opportunity for mentoring**
- **Functional requirements are important**
  - Basis for design
  - Basis for development
  - Basis for test scenarios

Requirements Definition



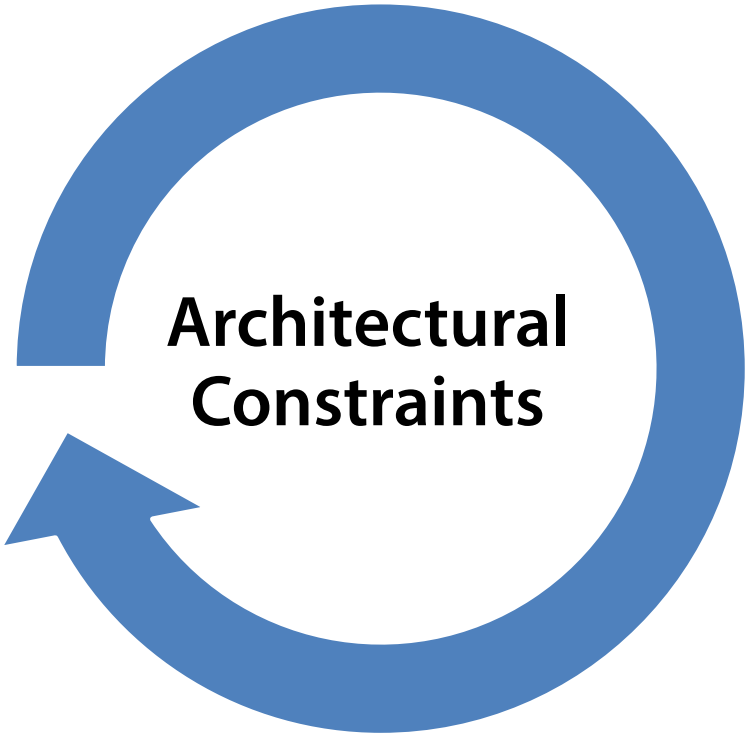
- Architect is a consumer of the requirements
- Provide support and guidance to BA team
- Architect must understand functional requirements

Requirements Definition



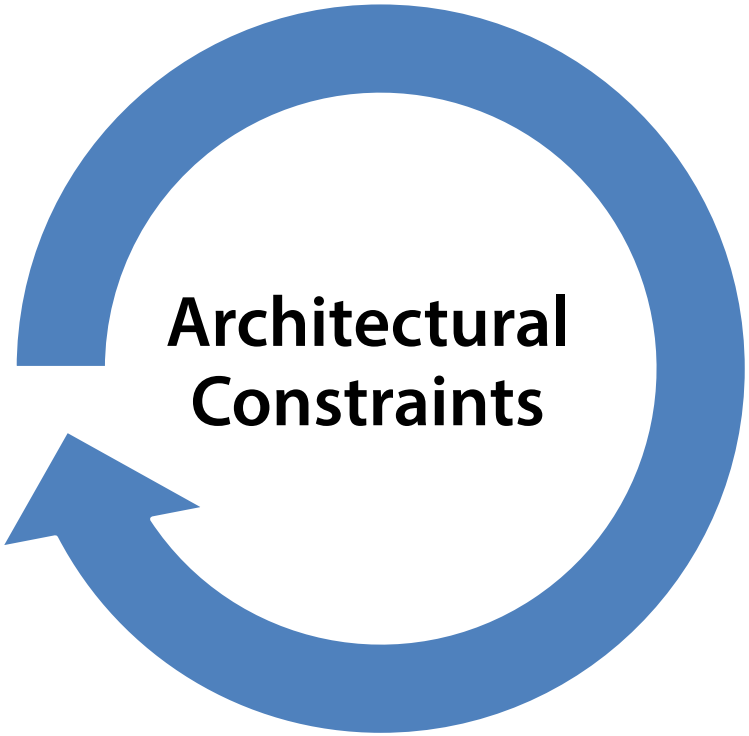
- Architectural constraints greatly influence design decisions
- Constraints impact design choices
- Constraints come in many flavors
- Architect identifies constraints

Requirements Definition



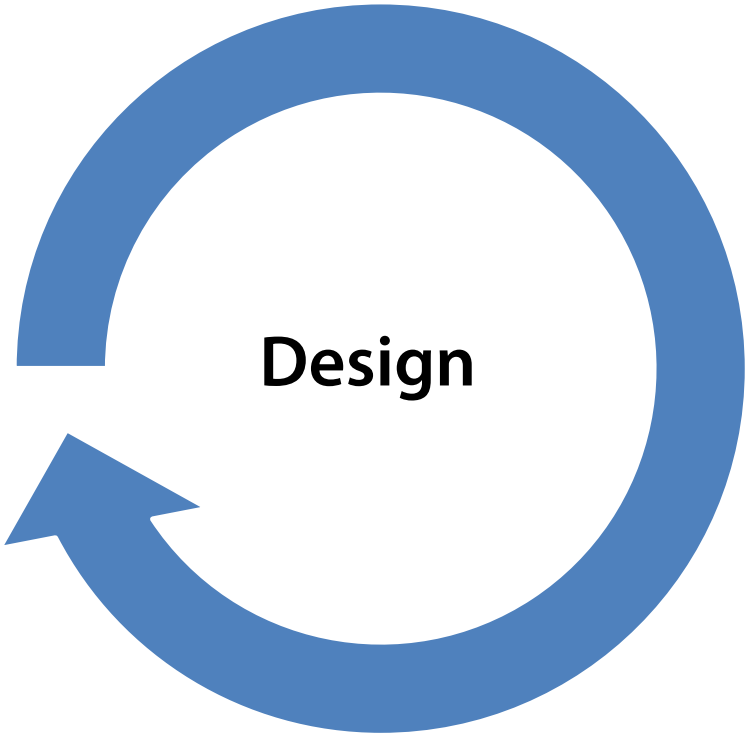
- **Examples:**
  - Time
  - Resource
  - Budget
  - Team Skills
  - Deployed infrastructure
  - Standards
    - Architectural
    - Technology
    - Coding

Requirements Definition



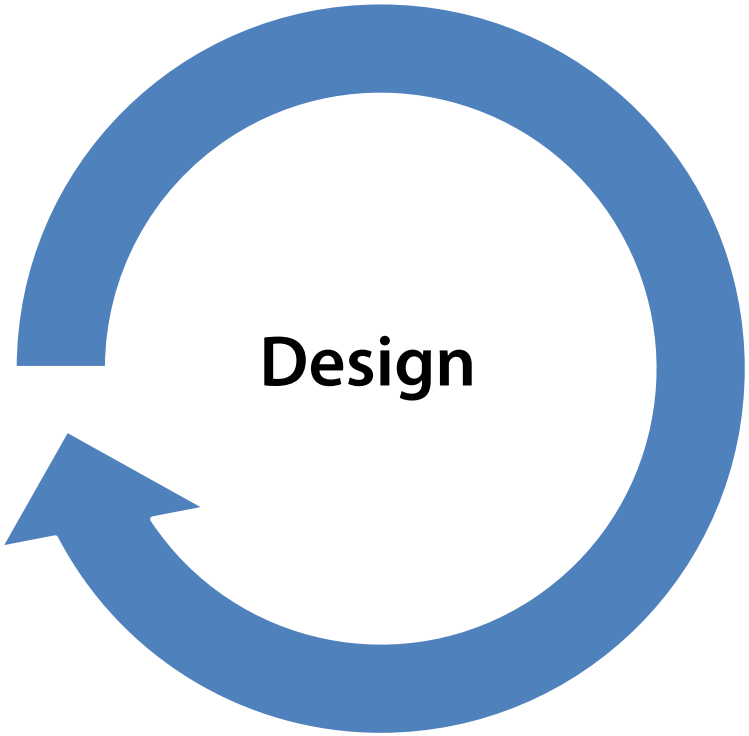
- **Come from customers, business and technology**
- **Required condition of design or implementation**
- **Architects job is to identify constraints**
  - Account for them in design
  - Account for them in implementation

The architect's role in the project life cycle



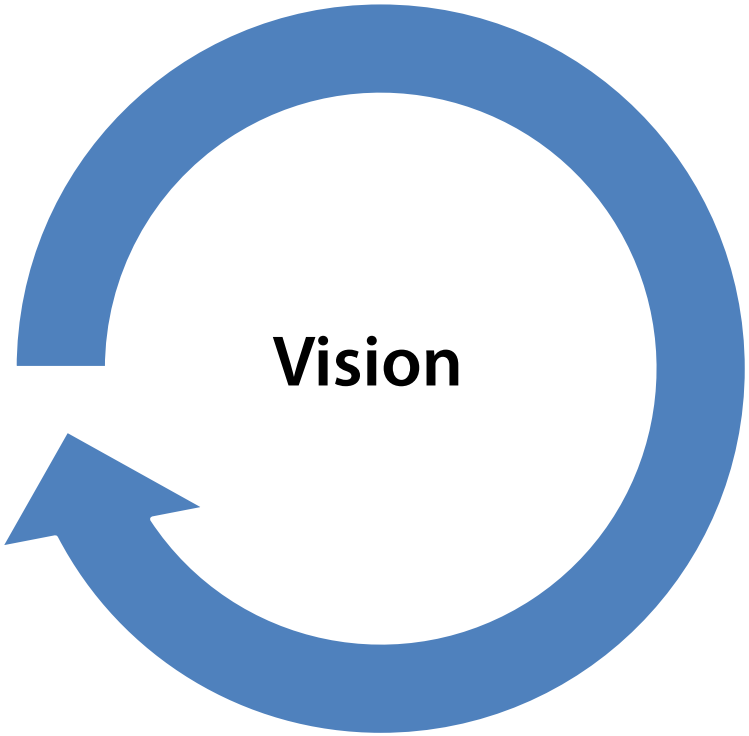
- Design is inward facing and creative
- Define structures that make up solution
- Documentation is outward facing and communicative
- Two distinct goals
- Design is the most interesting part of project for some people
- Design is creative
- Design solves complex problems with interesting solutions

Design



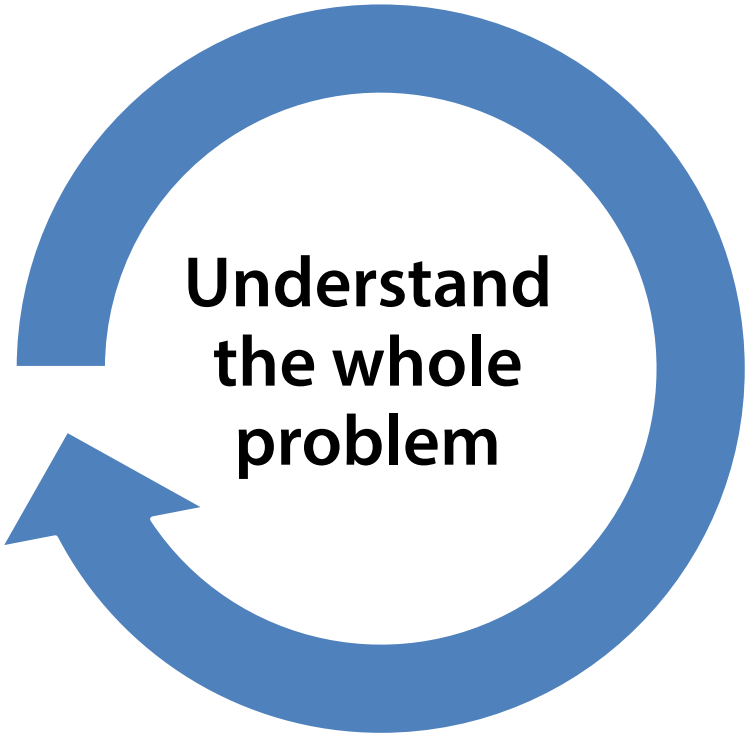
- **Least expensive time to make mistakes**
- **Make your mistakes during design**
  - Less expensive
- **Design is collaborative**
- **Design is iterative**
- **Trusted group of colleagues**
  - Joint design
  - Architectural review
  - Essential for creating solid designs

Design



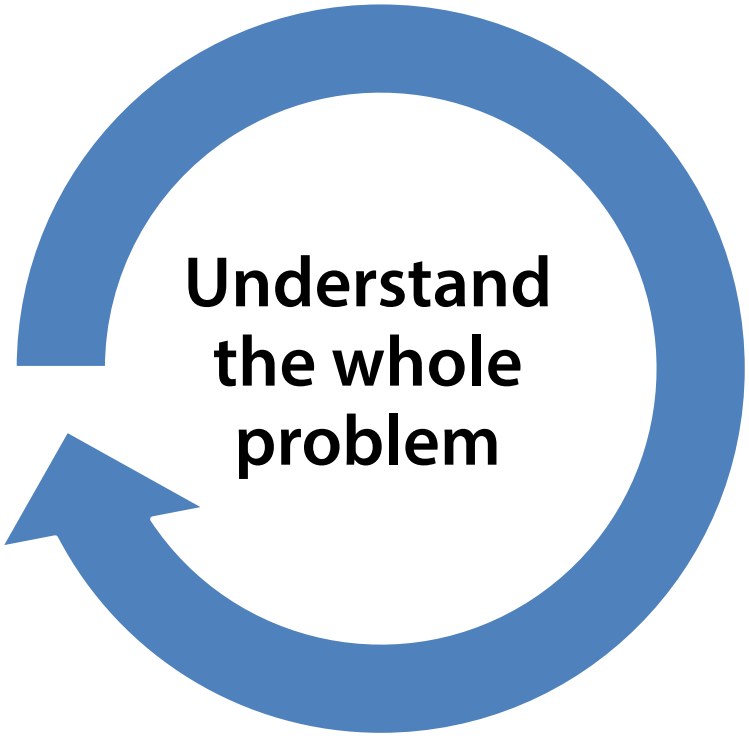
- Design is where we realize your visions true value
- Vision is the foundation of your design
- Design is iterative process of decomposition
- Vision guides the design at every level
- Make sure your vision is documented

Design



- **Process of understanding the whole problem will never be complete**
- **Paradox**
  - Must understand whole problem that is changing and evolving
- **Requirements change in every phase of the project**
  - Sometimes business changes
  - Sometimes we really don't understand the whole problem
- **Design is iterative**
- **Design must anticipate changes**
- **Modular designs help mitigate**

Design



- **How can I anticipate changes?**
  - You can't!
  - Identify risky parts of the solution
  - Design solution to mitigate these risks
- **Risky portions often occur at boundaries**
- **Pay careful attention to portions of the solution that you do not control**
  - Treat these as high risk
  - Protect the solution from risky dependencies
  - Insulate your solution from changes in these dependencies

Design

- **Leverage your technical feedback loop**
  - This team will help you identify design issues
  - Cultivate these relationships
- **Collaboration fosters ownership and support**
  - Advocates, evangelists & voice
  - Investment in solution
  - Collaborative design becomes teams
  - Help provide explanations for design decisions

Design

- **Small committed design teams are best**
- **3 people works best**
- **May collaborate with different people for different parts of the design**
  - High level design with one group
  - Detailed design with senior developers

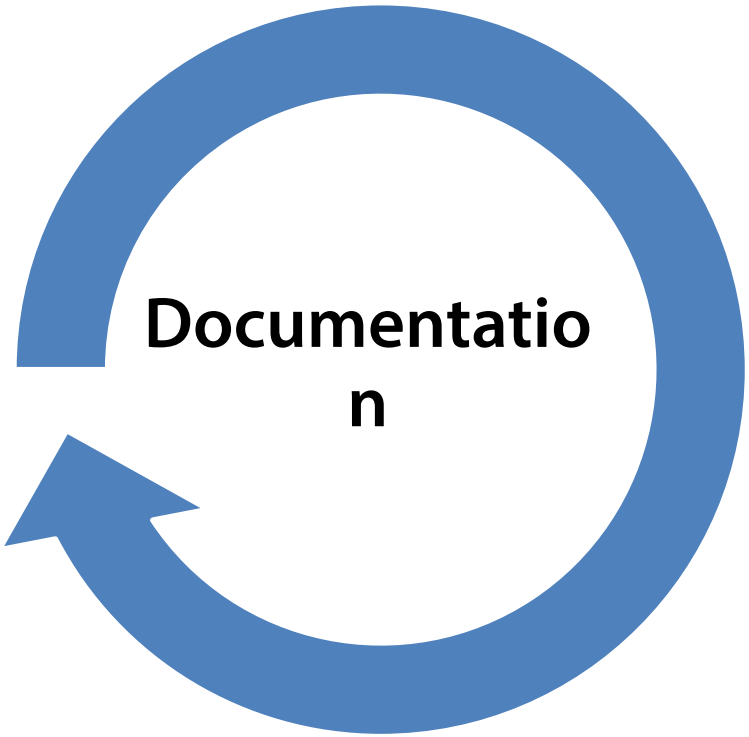
Design

- **When selecting your design team:**
  - Ideal candidates are experienced and senior developers and architects
  - Ideal candidates know how to offer constructive but not critical advice
  - Ideal candidates are not afraid to offer their frank opinion
  - This is not the place to mentor junior developers
  - This is not the place for unconstructive criticism

Design

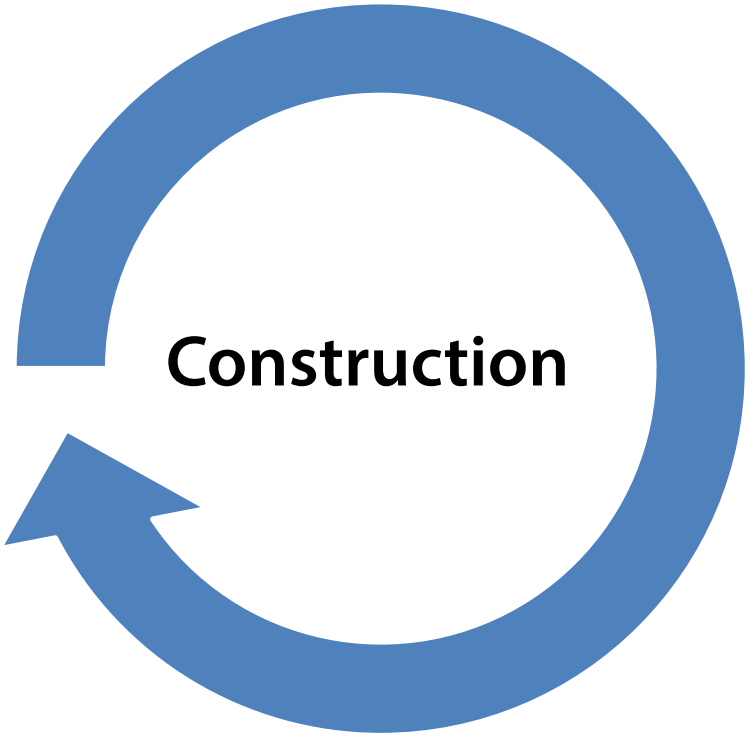
- **Design Sessions:**
  - Begin with brainstorming sessions
  - Architect will get everyone up to speed
  - Team must understand the whole problem that is being solved
  - Ideas should not be criticized
  - Identify the best ideas
  - Best ideas are often the synthesis of a few good ideas

Design



- Design identifies structures
- Views communicate structures to an audience
- Documentation communicates solution externally to an audience
- Views are representations of design to a particular audience
  - Some views target business
  - Some views target technical
  - Some views target both
- No right or wrong number of views
- View represents single perspective
- Views are windows into your design
- No single viewpoint represents entire solution

The architect's role in the project life cycle



- Technical team lead
- Guide the development team
- Assist with resource planning
- Fill technical gaps

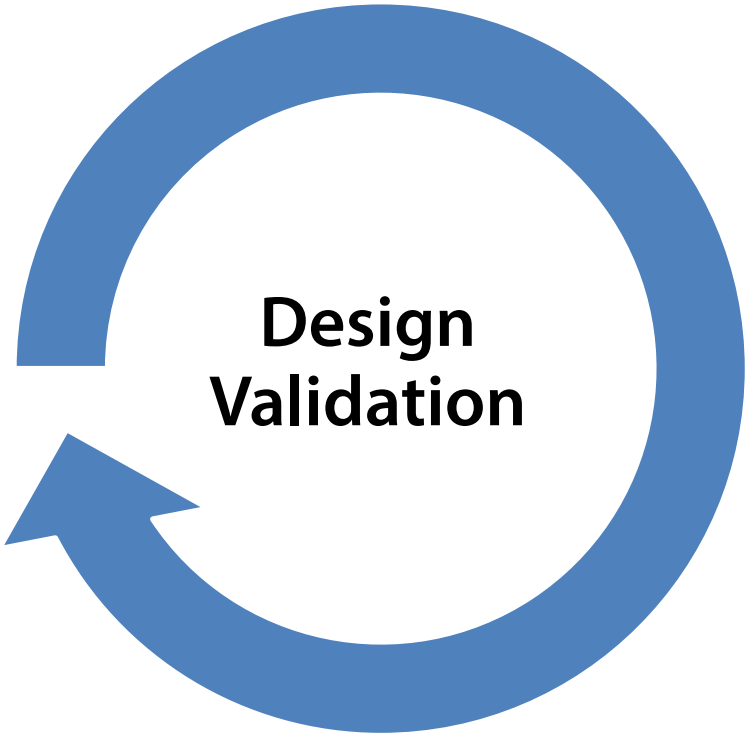
Construction

- Technical project manager
- Organize project into logical segments
- Resource planning
- Fills knowledge gap
- Train team

Construction

- Insulates development team from meeting overload
- Act as technical team representative
- Communicate decisions

Construction



- Stay one step ahead of development effort
- Reviews functional requirements
- Provide technical direction
- Architectural pivots
- Project scope negotiation
- Reacts to changes

Construction

- Establish development environment
- Environment should reflect production
- Separate from qa

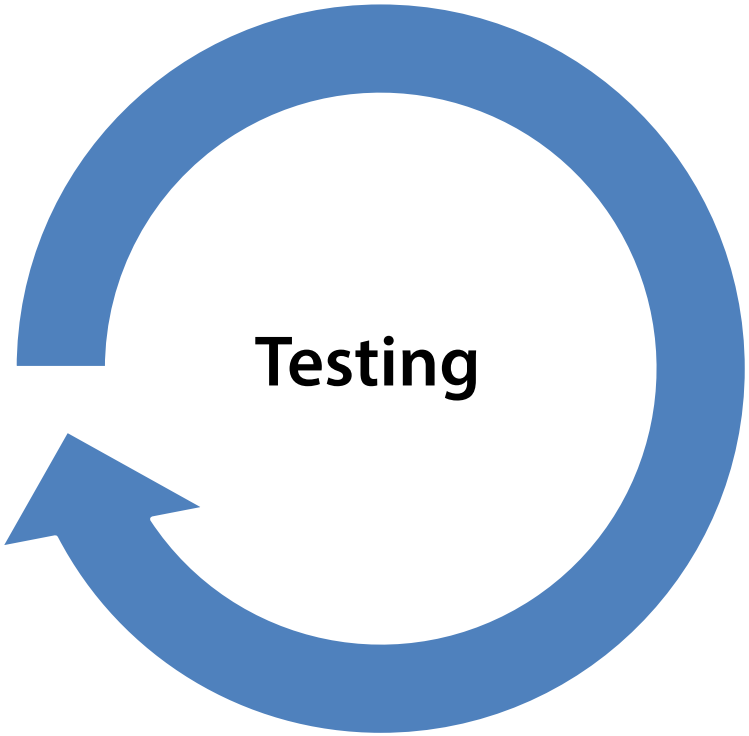
Construction

- Development IDE
- Programming Language
- Unit Testing requirements
- Change Management practices
- Source code repository
- Coding Standards
- Establish these standards if they don't exist

Construction

- **Code review should be a integral part of your process**
- **Required of every team member**
- **Open/team code reviews**
- **Goals:**
  - Verify design conformance
  - Insure code quality
  - Provide platform for developer to show work
  - Provide positive feedback
  - Motivate
  - Constructive criticism
  - Help team members grow
  - Train junior members
  - Educate entire team on solution
- **Positive experience**

The architect's role in the project life cycle

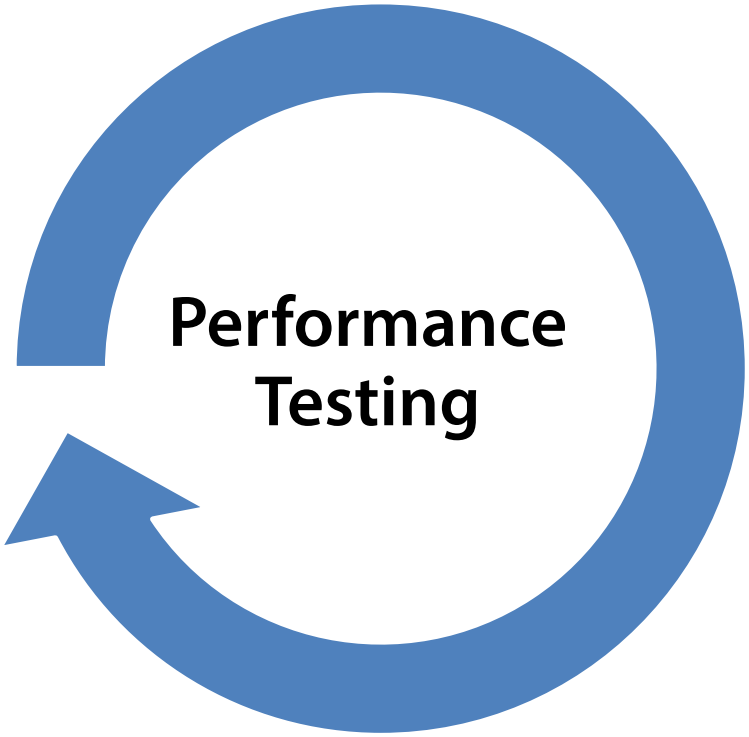


- Quality assurance testing
- Performance testing

Testing

- Supportive Role
- Help testers understand requirements
- Help translate requirements into test cases
- Make sure quality assurance environment exists
- Make sure quality assurance environment is being utilized
- Support development team
  - Advice
  - Strategies
  - Design changes

Testing

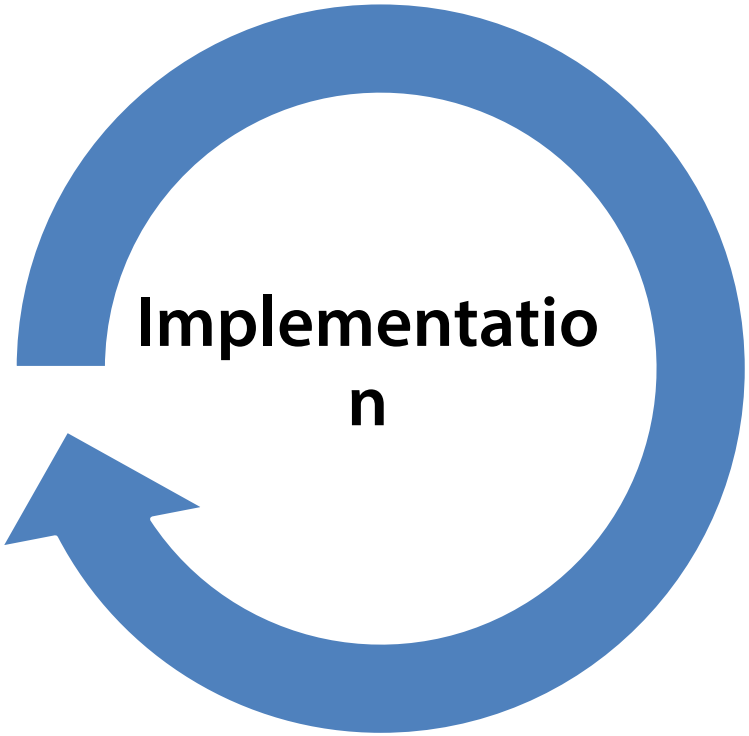


- Architect is actively involved
- Architect decides if performance testing is necessary for project
- Understand your quality constraints
- When performance testing is necessary
  - Support team
  - Guide team
- Architect may act as project manager for performance team
- Identify environment
- Identify performance testing tools

Testing

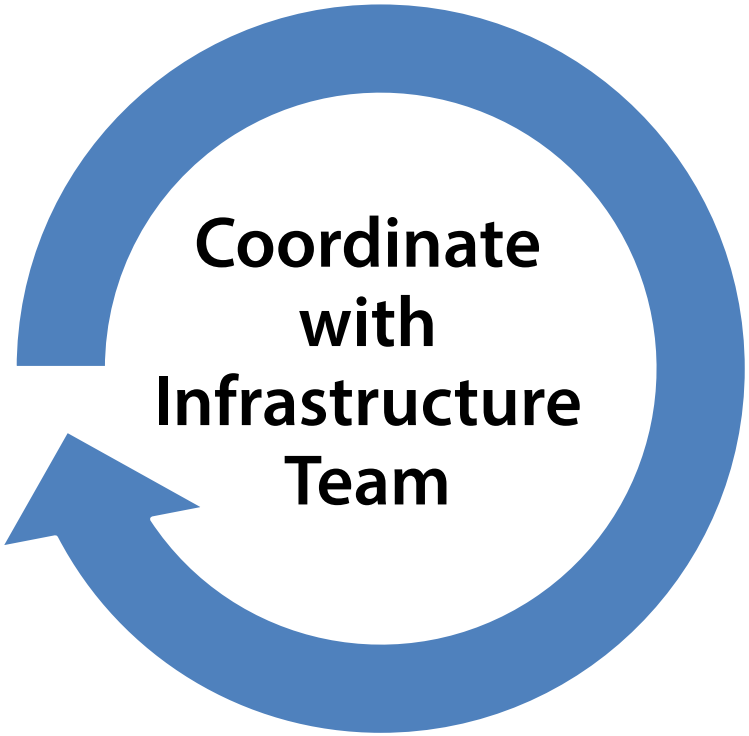
- **Guidance**
  - Acceptance criteria
  - Test design
  - Analyze test results
- **Performance related recommendations**
- **React to findings**
- **Modify architecture if needed**
- **Prototype**
- **Evaluate**

The architect's role in the project life cycle



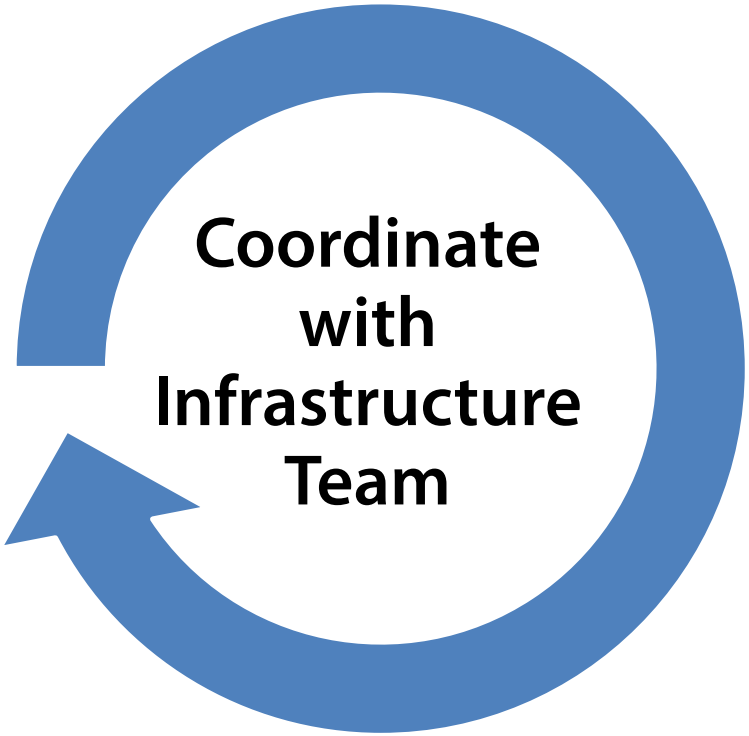
- **Coordinate deployment**
  - Change management team
  - Infrastructure team

Implementation



- **New or existing infrastructure?**
- **Existing infrastructure should be identified as an architectural constraint**
- **New environment**
  - Collaborate with infrastructure
  - Physical tier design
- **Implementation considerations**
  - Availability
  - Interoperability
  - Performance
  - Reliability
  - Scalability
  - Security
- **Non-functional run-time attributes may greatly impact implementation**

Implementation

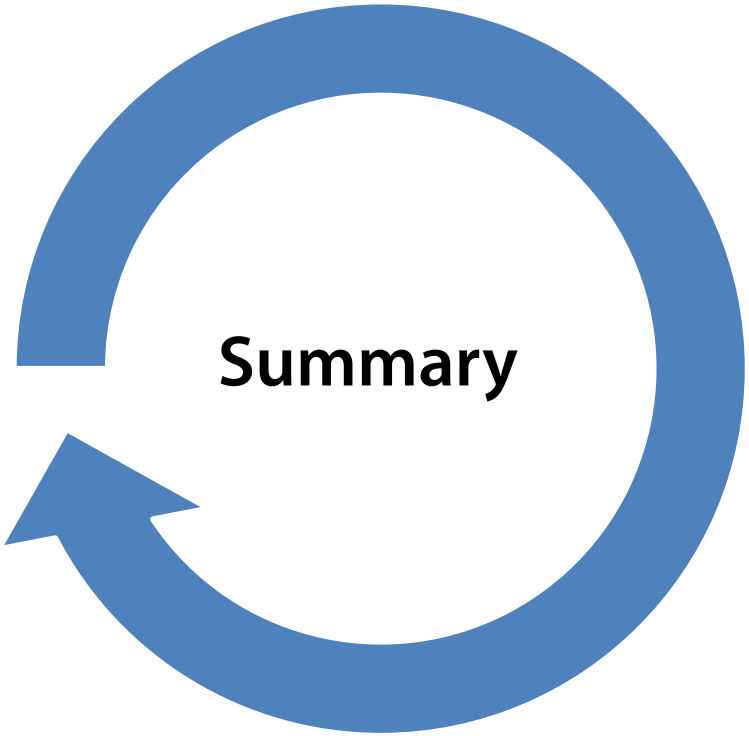


- Heath Check
- Performance Monitors

Implementation

- **Collaborate, mentor & guide change management team**
- **3 Main areas of concern:**
  - Source code management
  - Build management
  - Deployment
- **Architect must ensure source control is in place**
- **Architect must ensure automated build process is in place**
- **Coordinate solution deployment to production**
- **Architect must ensure automated deployments are in place**
- **Architect remains available for troubleshooting if needed**

The architect's role in the project life cycle



- Duties of the software architect in each phase of the project
- Effective architects don't just design solutions
- Architects play an important role from initiation through implementation
- Active participants in each and every phase of the project
- May be a bit daunting
- Representation of the duties performed
- Architect role is very opaque
- Varies between organizations
- Many organizations don't have clearly defined roles for the architect
- Up to you to define your role in the